Satisfiability checking and application of Gröbner bases

Eva María González García

Master in Mathematics

University of Granada

2021/2022

Master Thesis

Satisfiability checking and application of Gröbner bases

Eva María González García

Advisor: Prof. Dr. D. Pascual Jara Martínez

Master in Mathematics
University of Granada
2021/2022

Contents

Introduction

Given a logic $\mathcal{L}$ and a formula of that logic, the *decision problem*, also called *satisfiability problem*, tries to answer whether that formula is valid or equivalently if it is satisfiable. Then, a *decision procedure* for the logic $\mathcal{L}$ is an algorithm that solves the decision problem for any formula in $\mathcal{L}$. We will say that a decision procedure is *sound* if it always returns the correct answer, and it is *complete* if it answers in a finite time. A *decidable* logic is one for which there exists a sound and complete decision procedure. In propositional logic, the problem consists in determining if there is a truth assignment of its variables for which the formula is true. In the worst case, checking satisfiability involves testing 2^n inputs, where n is the number of variables in the formula. This is the first problem that was proven NP-complete in 1971 by Cook [16]. It is called the boolean satisfiability problem also called *SAT*. First order logic is more expressive than propositional logic. However, the satisfiability problem for first order logic is undecidable, since there is no algorithm that returns a correct answer in a finite time for the satisfiability problem of any first order formula. In spite of this, there are fragments of first order theories that are decidable. That motivates the so-called *satisfiability modulo theories* problem (SMT).

Due to the continuous improvement of SAT solving technology, it turns out a good decision to use SAT solvers within SMT solvers. It is the case of the eager and lazy approaches of SMT solving. The *eager approach* to SMT solving involves translating the original formula into an equisatisfiable propositional formula, while the *lazy approach* uses SAT solvers along with decision procedures for first order theories called theory solvers. These *theory solvers* are dedicated methods tailored to the specific theory. In the image 1 we can see in broad outline the scheme that follows a lazy approach of SMT solver. The input is usually a first order formula in conjunctive normal form. As we can see, in the image the input is of the form

$$(P_{11} \wedge \cdots \wedge P_{1n_1}) \vee \cdots \vee (P_{m1} \wedge \cdots \wedge P_{mn_m})$$

where P_{ij} are predicates applied to the corresponding terms. When the SAT solver receives the formula, it builds a propositional formula that is nothing more than the same formula where the predicates have become propositional variables that can take values `true` or

`false`. Afterwards, the SAT solver search for a satisfying assignment, if it does not find such assignment, it returns `UNSAT`.

$$(P_{11} \wedge \cdots \wedge P_{1n_1}) \vee \cdots \vee (P_{m1} \wedge \cdots \wedge P_{mn_m})$$

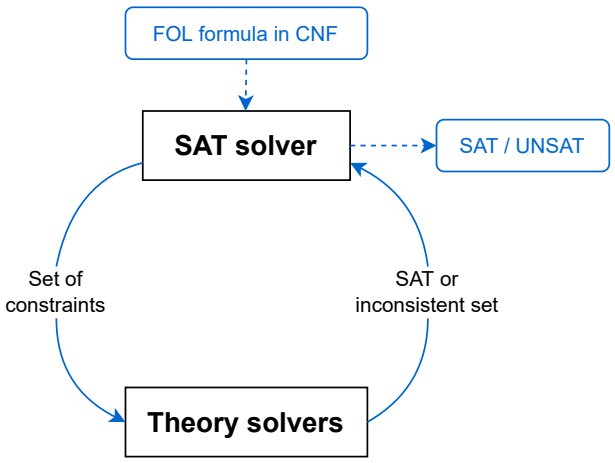


Figure 1: Diagram of lazy SMT solvers

In case the SAT solver does find an assignment, then the set of predicates associated with the propositional variables that were assigned `true`, is sent to the corresponding theory solver. The theory solver check the consistency of the predicates within the theory, that is, whether there exists an assignment to the first order variables that makes the predicates `true`. If so, then the system returns `SAT`. If the set of predicates is not consistent within the theory, the theory solver returns a set of inconsistent predicates to the SAT solver, which generates a conflict clause with the propositional variables that correspond to the conflicting predicates. This clause is used so that the SAT solver does not generate another assignment that assigns `true` to predicates in the conflicting set any more.

Recently, satisfiability checking and symbolic computation communities have found that some of their lines of research have started to converge. Symbolic computation usually refers to the study and development of algorithms and software for manipulating symbolic objects such as algebraic expressions, logical propositions and programs themselves. Computer algebra systems have functionality for Gröbner basis, cylindrical algebraic decomposition, graph algorithms, etc. The use of computers to tackle mathematical problems dates back to 1953, being almost as old as computing itself. It is longer than that of satisfiability checking, appearing the first SAT solver that follows the modern design in the 1990s. The first algebraic software appeared in the 1960s with new methods for factoring polynomials.

Buchberger's development of the Gröbner bases was a breakthrough in the field. Symbolic computational systems solve abstract problems, but while computer algebra systems are not optimized for problems that require searching through a large combinatorial space, this is something that SAT solvers excel at.

A combination of SAT solvers and symbolic computation is used to solve conjectures in this paper [15]. Symbolic computation can also be used to solve the boolean satisfiability problem by means of Gröbner bases computation, as discussed in [19]. Also, an example of the combination of those two fields are SMT solvers for the theory of non-linear real arithmetic [1, 3]. A decision procedure for the last example is the cylindrical algebraic decomposition [5]. Cylindrical algebraic decomposition was introduced by George Collins in 1975 and has much better computational complexity than the initial version, quantifier elimination, developed by Tarski in the 1930s. The quantifier elimination decision procedure proved that non-linear real arithmetic is decidable.

Despite the benefits that both communities would obtain from the collaboration between them, their research areas are still very disconnected. Furthermore, their libraries can not be easily embedded since it requires either a deep understanding of complex mathematical problems or efficient solver implementations. To support a stronger collaboration, several initiatives have been launched. SC^2 was created to bridge these two communities, they organize some community building events such us workshops and summer schools [2]. In 2016, SC^2 obtain funding from the European Commission. SMT-LIB states common standards and a library of benchmarks for the comparison of SMT systems. This community also work to extend SMT-LIB to better support the areas of relevance to SC^2.

In this thesis, we are focused on how to solve a few satisfiability problems using Gröbner bases. The computation of the Gröbner basis of a system of non-linear multivariate polynomials can be seen as a term rewriting method, since it allows us to obtain a simpler system of equations that represents the same set of solutions as the initial system [7]. Once the Gröbner basis is calculated, we can check if the set of equations has any solutions using Hilbert's Nullstellensatz theorem. This theorem states that to check this, it is only necessary to check whether 1 belongs to the Gröbner basis.

In the first application, Gröbner bases are used to define a SAT solver by algebraic methods. The propositional formula is converted into a polynomial over the finite field $\mathbb{F}_2$ where $0, 1$ represent `true`, `false` respectively. After, the Gröbner basis of a system of equations is computed and the problem is reduced to the verification of whether 1 belongs to the Gröbner basis. A simple implementation of an algebraic SAT solver is presented at the end.

The other application is the resolution of the existential fragment of Presburger's theory of arithmetic. Using a lazy SMT solver, as we saw in the image 1, the problem is reduced

to checking if a set of linear equations defined over the natural numbers has a common solution, that is to check if there is an assignment that satisfies those constraints. As the set of natural numbers is not algebraically closed, we can not compute the Gröbner basis of this system of equations directly.

Then, we will prove the Herbrand's expansion theorem, which will be used to prove that the satisfiability problem in first order logic is undecidable and satisfiability in propositional logic is NP-complete. In a next chapter, we review the Gröbner bases theory. We will use this tool to solve two satisfiability problems.

Chapter I

Satisfiability

A **SAT solver** is a decision procedure for *propositional logic*, while **SMT solvers** aims at determining the satisfiability of *first order logic* formulas regarding some background theories for which the decision problem is decidable. To understand in more depth what SAT and SMT consist of, in this chapter we will discuss the following topics whose content has been taken from the specified references.

1. Review of Propositional Logic (PL) [14, 29, 28].
2. Review of First Order Logic (FOL) [14, 32, 36].
3. Determination of complexity of the satisfiability problem for propositional logic and first order logic [32, 36, 38, 23, 18].
4. Introduction to first order theories and fragments and some examples [14, 19].

1 Propositional logic

We will start by illustrating with an example what a propositional formula is. Let's say we have two numbers a and b, the statement *"a is greater than b"* would be a variable of PL which can be `true` or `false` but not both at the same time.

A more complex example which uses connectives is *"if a is greater than b and b is greater than c, then a is greater than c"*. If we define the variables $A =$ *"a is greater than b"*, $B =$ *"b is greater than c"* and $C =$ *"a is greater than c"*, the corresponding logic formula is $(A \wedge B) \rightarrow C$.

Syntax

The elements of propositional logic are:

- **truth values**: 1 - `true`, 0 - `false`.
- **atoms** are the propositional variables $A, B, C, \ldots$. We can assign truth values to the atoms. The set of atoms is a countably infinite set $\mathcal{A}_P$.
- **connectives** can be applied to atoms or formulas. A d-ary connective is of the form $C : \{0,1\}^d \to \{0,1\}$. The set of connectives is denoted by $\mathcal{C}$.

Example. 1.1.
$\neg$ and id are examples of 1-ary connectives, while $\wedge$, $\vee$, $\to$ $\leftrightarrow$, and $\oplus$ are 2-ary connectives.

a	$id(a)$	$\neg a$
0	0	1
1	1	0

a	b	$a \wedge b$	$a \vee b$	$a \to b$	$a \leftrightarrow b$	$a \oplus b$
0	0	0	0	1	1	0
0	1	0	1	1	0	1
1	0	0	1	0	0	1
1	1	1	1	1	1	0

Figure I.1: Truth tables of id, $\neg$, $\wedge$, $\vee$, $\to$, $\leftrightarrow$, and $\oplus$.

A **proposition** is constructed by applying connectives to one or several atoms or propositions. If $\mathcal{P}_1 = \{C(P_1, \ldots, P_d) : C \in \mathcal{C}, P_i \in \mathcal{A}\}$ is the set of connectives applied to the variables, then $\mathcal{P}_n = \{C(P_1, \ldots, P_d) : C \in \mathcal{C}, P_i \in \mathcal{P}_{n-1}\}$ is a set of propositions. As we assume that $id \in \mathcal{C}$, then $\mathcal{P}_n \subset \mathcal{P}_{n+1}$ for all $n \in \mathbb{N}$.

The set $\mathcal{L}_P = \bigcup_{n\in\mathbb{N}} \mathcal{P}_n$ contains the propositions that can be formed by the elements of the propositional logic.

Semantics

An **interpretation** $\nu : \mathcal{A}_P \to \{0,1\}$ is a function that assigns to every propositional variable a truth value. We can extend this definition to the set of propositions $\nu : \mathcal{L}_P \to \{0,1\}$, in this way for each $P = C(P_1, \ldots, P_d) \in \mathcal{L}_P$ where $C \in \mathcal{C}$,

$$\nu(P) = \nu(C(P_1, \ldots, P_d)) = C(\nu(P_1), \ldots, \nu(P_d)).$$

This is the result of assigning the corresponding values to each atom of the proposition.

Satisfiability

A proposition $P \in \mathcal{L}_P$ is **satisfiable** if there is an interpretation ν such that $\nu(P) = 1$. Otherwise, it is **unsatisfiable**. It is **valid** if for all ν interpretation, $\nu(P) = 1$.

Example. 1.2.

- $A \vee \neg A$ is valid.
- $A \wedge \neg A$ is unsatisfiable.
- $A \wedge \neg B$ is satisfiable, and we check it by using the interpretation $\nu : \{A \mapsto 1, B \mapsto 0\}$ since $\nu(A \wedge \neg B) = \nu(A) \wedge \nu(\neg B) = \nu(A) \wedge \neg \nu(B) = 1 \wedge 1 = 1$.
- $(\neg A \vee B) \wedge (A \vee \neg B \vee C)$ is satisfiable for the interpretation $\nu : \{A \mapsto 0, B \mapsto 0, C \mapsto 1\}$.
- $(\neg A \vee B) \wedge (A \vee B) \wedge \neg B$ is unsatisfiable.

Lemma. 1.3.
Given $P \in \mathcal{L}_P$, P is valid if, and only if $\neg P$ is unsatisfiable.

PROOF.

$\Rightarrow$) If P is valid, then for any interpretation ν, $\nu(P) = 1$. Equivalently, $\nu(\neg P) = 0$. Then, $\neg P$ is unsatisfiable.

$\Leftarrow$) If $\neg P$ is unsatisfiable, then for any interpretation ν, $\nu(\neg P) = 0$. Equivalently, $\nu(P) = 1$. Then, P is valid. □

Normal forms

A **literal** is a variable or its negation. The connectives $\wedge$ y $\vee$ are called **conjunction** and **disjunction**, respectively. A **clause** is a disjunction of literals or a single literal.

A formula in **conjunctive normal form (CNF)** is a conjunction of clauses $\bigwedge_i \bigvee_j l_{i,j}$ for literals $l_{i,j}$.

Example. 1.4.
The formula $(A \vee B \vee C) \wedge (D \vee B \vee C)$ is in conjunctive normal form.

A formula in **disjunctive normal form (DNF)** is a disjunction of conjuctions $\bigvee_i \bigwedge_j l_{i,j}$ for literals $l_{i,j}$.

Example. 1.5.
The formula $(A \wedge B) \vee (A \wedge C) \vee (D \wedge B) \vee (D \wedge C)$ is in disjunctive normal form.

2 First order logic

Compared to PL, we have a higher level of freedom in FOL. Considering the statement we used before *"a greater than b"* (an atom in PL), we create the predicate $>(x, y)$ that is equivalent to $x > y$. Now, x and y are variables, and depending on the value they take, the predicate is `true` or `false`. It is also possible to define the formula with quantifiers $\forall x \exists y . x \geqslant y$, or define the function $+(x, y)$, that represents $x + y$, and write the formula $\forall x . x + d > x$ where d is a constant. This makes first order logic more expressive than the propositional logic.

Syntax

The elements of first order logic are:

- **variables** $x, y, z, \ldots$ and **constants** $a, b, c, \ldots$ which are the most basic **terms**. The set of variables and constant symbols is denoted by $\mathcal{V}$ and the set of terms by $\mathcal{T}$.
- **functions** $f, g, h, \ldots$ are used to construct more complex terms. A d-ary function is of the form $f : \mathcal{T}^d \to \mathcal{T}$. The set of function symbols is denoted by $\mathcal{M}$.
- **predicates** $A, B, C, \ldots$ are the atoms of first order logic. A d-ary predicate is of the form $P : \mathcal{T}^d \to \{0, 1\}$. Predicates are seen as relations where $(t_1, \ldots, t_d) \in P$ if $P(t_1, \ldots, t_d) = 1$. The set of predicates symbols is denoted by $\mathcal{N}$.
- in addition to the elements of the PL, there are two quantifiers:
  - **universal quantifier**: $\forall x.x + d > x$ means *"for all x, $x + d > x$"*;
  - **existential quantifier**: $\exists x.f(x) = 0$ means *"there exists x such that $f(x)$ is equal to 0"*.

  Quantifiers are denoted by $\mathcal{Q} \in \{\exists, \forall, \varnothing\}$, where $\varnothing$ means that no quantifier is applied.

A **term** is constructed by applying functions to one or several variables, constants or terms. Let $\mathcal{T}_1 = \{f(t_1, \ldots, t_d) : f \in \mathcal{M}, t_i \in \mathcal{V}\}$ be the set of functions applied to the variables and constants, then $\mathcal{T}_n = \{f(t_1, \ldots, t_d) : f \in \mathcal{M}, t_i \in \mathcal{T}_{n-1}\}$ is a set of terms. We convene that some function symbol is the identity function, then $\mathcal{T}_n \subset \mathcal{T}_{n+1}$ for all $n \in \mathbb{N}$.

- The set $\mathcal{T} = \bigcup_{n\in\mathbb{N}} \mathcal{T}_n$ contains the terms that can be obtained by using constant, variable, and function symbols.
- The set $\mathcal{A}_{FO} = \{P(t_1, \ldots, t_d) : P \in \mathcal{N}, t_1, \ldots, t_d \in \mathcal{T}\}$ contains the predicates applied to the terms.

A **formula** is constructed by applying connectives or quantifiers to one or several atoms or formulas. If $\mathcal{F}_1 = \{\mathcal{Q}x.C(F_1, \ldots, F_d) | C \in \mathcal{C}; x, F_i \in \mathcal{A}_{FO}\}$ is the set of connectives applied to the variables, then $\mathcal{F}_n = \{\mathcal{Q}x.C(F_1, \ldots, F_d) | C \in \mathcal{C}; x, F_i \in \mathcal{F}_{n-1}\}$ is a set of propositions. Since $id \in \mathcal{C}$, then $\mathcal{F}_n \subset \mathcal{F}_{n+1}$ for all $n \in \mathbb{N}$.

The set $\mathcal{L}_{FO} = \bigcup_{n\in\mathbb{N}} \mathcal{F}_n$ contains the formulas that can be formed by the elements of first order logic.

A variable can be classified regarding the state of that variable within a specified formula:

- **free**: there is an occurrence of x that is not bound by any quantifier,
- **bound**: there is an occurrence of x in the scope of a binding quantifier $\forall x$ or $\exists x$.

Example. 2.1.
In the formula

$$F = \forall x.p(f(x), y) \rightarrow \forall y.p(f(x), y),$$

x only occurs bound, and y appears as free (in the left-hand side of the implication) and bound (in the right-hand side). Observe that F is the scope of the quantifier $\forall x$ and $p(f(x), y)$ is the scope of the quantifier $\forall y$.

A formula is **closed** if it does not contain any free variables.

Semantics

First order formulas evaluate to truth values, but terms evaluates to values from a specific domain. An **interpretation** $\nu : (D_\nu, \alpha_\nu)$ consisting of a domain and an assignment. A **domain** D_ν of an interpretation ν is a non-empty set, where examples of elements of this set are the integer and real numbers, or any other physical or abstract objects. Domains can be finite, countably or uncountable infinite.

The **assignment** α_ν of an interpretation ν is a map:

- each variable or constant is assigned a value from D_ν;
- each n-ary function symbol is assigned an n-ary function $f_\nu : D_\nu^n \rightarrow D_\nu$;
- each n-ary predicate symbol P is assigned an n-ary predicate $P_\nu : D_\nu^n \rightarrow \{\texttt{true},\texttt{false}\}$;

A formula $F \in \mathcal{L}_{FO}$ is interpreted as follows:

- $\nu(x) = x_\nu$ for every free variable or constant x in F,
- $\nu(S(s_1, \ldots, s_d)) = S_\nu(\nu(s_1), \ldots, \nu(s_d))$ for every symbol S in F such that $S \in \mathcal{C} \cup \mathcal{N} \cup \mathcal{M}$,
- $\nu(\forall x.F) = 1$ if and only if $[\forall a \in D_\nu : \nu(x) = a]\nu(F) = 1$,
- $\nu(\exists x.F) = 1$ if and only if $[\exists a \in D_\nu : \nu(x) = a]\nu(F) = 1$.

A **model** of F is and interpretation ν such that $\nu(F) = 1$.

Example. 2.2.
The formula $F = \forall y \exists x.P(y,x) \wedge \exists z \forall w.\neg P(w,z)$ consists of the binary predicate symbol P, and the variables x, y, z, w.

We construct an interpretation $\nu : (D_\nu, \alpha_\nu)$ where $D_\nu = \mathbb{N}$. The predicate P is assigned the less-than relation $<$. The variables x, y, z, w are assigned the values $13, 12, 0, 4$, respectively. Then we obtain the interpretation $\nu : (\mathbb{N}, \alpha_\nu)$, where

$$\alpha_\nu : \{P \mapsto <, x \mapsto 13, y \mapsto 12, z \mapsto 0, w \mapsto 4\}.$$

All the other constant, function and predicate symbols that do not appear in F are ignored. Then, the formula becomes

$$\exists x.(y < x) \wedge \exists z \forall w.\neg(w < z).$$

Satisfiability

A formula is satisfiable if it has a model. It is valid if every interpretation is a model. Normally, satisfiability only apply to closed formulas, but there is convention, a formula F is satisfiable if its existential closure $\exists *.F$ is satisfiable, it is valid if its universal closure $\forall *.F$ is valid.

Normal forms

Two formulas F and G are **equivalent** if for all interpretation ν, then $\nu(F) = \nu(G)$. It is denoted by $F \equiv G$. They are **equisatisfiable** when, F is satisfiable if and only if G is satisfiable.

A formula is **rectified** if no variable occurs both bound and free, and if all the quantifiers in the formula have an effect on different variables.

The rectified equivalent formula is computed by renaming variables, making the bounding variables disjoint for different subformulas. Given a formula F, a variable in this formula x and a term t which does not appear in F, $F[x/t]$ is the result of replacing any free occurrence of x by t. It is also used $F[x_1/t_1, \ldots, x_n/t_n]$ for the replacement to be simultaneously, it should not be confused with $F[x_1/t_1] \cdots [x_n/t_n]$. To replace a bound variable, the equality $QxF \equiv QtF[x/t]$ is applied.

Example. 2.3.

1. $P(x)[x/y, y/z] = P(y)$. Conversely, $P(x)[x/y][y/z] = P(y)[y/z] = P(z)$.

2. In $F = R(x) \vee \forall x P(x,x)$, the variable x is both free and bound. To get the rectified formula, we select a new variable y and replace the bound occurrences of x. The resulting rectified formula is

$$R(x) \vee \forall x P(x,x) \equiv R(x) \vee \forall y P(x,x)[x/y] = R(x) \vee \forall y P(y,y).$$

3. In $F = \exists x(S(x) \wedge (\forall y P(x,y) \vee \neg y S(y))) \wedge R(x)$, x is both free and bound, and there are two occurrences of $\forall$ with the variable y. To get the rectified form, we select the variables w, z:

$$\exists w(S(x) \wedge (\forall y P(x,y) \vee \neg \forall z S(y)[y/z]))[x/w] \wedge R(x) \\ = \exists w(S(w) \wedge (\forall y P(w,y) \vee \neg \forall z S(z))) \wedge R(x).$$

Lemma. 2.4.
Any formula is equivalent to a rectified formula.

A rectified formula is in **prenex form** (RPF) if it has the form $Q_1 y_1 \cdots Q_n y_n . F$, where $Q_i \in \{\exists, \forall\}$, $n \geqslant 0$, and F is quantifier-free. Any rectified formula can be expressed in prenex form by using these equivalences:

- For $F \in \mathcal{L}_{FO}$ and variable x:

$$\neg \exists x F \equiv \forall x \neg F,$$
$$\neg \forall x F \equiv \exists x \neg F.$$

- For $F, G \in \mathcal{L}_{FO}$, the variable x has no free occurrence in G, since it is a rectified formula:

$$(\exists x F \vee G) \equiv \exists x (F \vee G),$$
$$(\exists x F \wedge G) \equiv \exists x (F \wedge G),$$
$$(\forall x F \vee G) \equiv \forall x (F \vee G),$$
$$(\forall x F \wedge G) \equiv \forall x (F \wedge G).$$

Example. 2.5.
In this example we express in prenex form the rectified formulas of the Example (2.3.):

1. The rectified formula $F = R(x) \vee \forall y P(y, y)$ is equivalent to $\forall y(R(x) \vee P(y, y))$.
2. Moving $\exists w$ to the left in the rectified formula $\exists w(S(w) \wedge (\forall y P(w, y) \vee \neg \forall z S(z))) \wedge R(x)$, we get
$$\exists w((S(w) \wedge (\forall y P(w, y) \vee \neg \forall z S(z))) \wedge R(x)).$$
After applying $\neg \forall z S(z) \equiv \exists z \neg S(z)$,
$$\exists w((S(w) \wedge (\forall y P(w, y) \vee \exists z \neg S(z))) \wedge R(x)).$$
Finally, by moving to the left in several steps $\forall y \exists z$, we obtain the formula in prenex form
$$\exists w \forall y \exists z((S(w) \wedge (P(w, y) \vee \neg S(z))) \wedge R(x)).$$

Lemma. 2.6.
Any formula is equivalent to a rectified formula in prenex form (RPF).

The **Skolem form** of a rectified formula in prenex form is a formula that does not contain any existential quantifier. It can be computed as follows:

SKOLEMIZATION_ALGORITHM(F)**:**

Input: $F = \forall y_1 \cdots \forall y_n \exists x_1 \cdots \exists x_m G$ in RPF

Result: Skolem normal form of F

for $\exists x_i$ in F:

take $f_{x_i}(y_1, \ldots, y_n)$ an n-ary function that does not appear in F

$F = \forall y_1 \cdots \forall y_n \exists x_{i+1} \cdots \exists x_m G[x_i / f_{x_i}(y_1, \ldots, y_n)]$

return F

Example. 2.7.
The Skolem form of the formula $F = \forall y_1 \forall y_2 \exists x_1 \exists x_2 (P(y_1) \vee (\neg R(x_1, y_1, y_2) \wedge R(x_2, y_1, y_2)))$ is:

$$\forall y_1 \forall y_2 (P(y_1) \vee (\neg R(f_{x_1}(y_1, y_2), y_1, y_2) \wedge R(f_{x_2}(y_1, y_2), y_1, y_2))).$$

Lemma. 2.8.
Any formula and its Skolem form are equisatisfiable.

PROOF. From Lemma (2.6.), any formula can be expressed in rectified prenex form and both are equivalent. As a result, we obtain the formula

$$F = \forall y_1 \cdots \forall y_n \exists x_1 \cdots \exists x_m G.$$

Using the algorithm to convert F in its Skolem form, we get the formula

$$F' = \forall y_1 \cdots \forall y_n G[x_1/f_{x_1}(y_1, \ldots, y_n), \ldots, x_m/f_{x_m}(y_1, \ldots, y_n)].$$

We are going to prove that F is satisfiable if and only if F' is satisfiable.

$\Leftarrow$) Assume there exists ν such that $\nu(F') = 1$. Then, $\forall b_1, \ldots, b_n \in D_\nu$,

$$\nu_{[y_1/b_1, \ldots, y_n/b_n]}(G[x_1/f_{x_1}(y_1, \ldots, y_n), \ldots, x_m/f_{x_m}(y_1, \ldots, y_n)]) = 1.$$

Thus, $\nu_{[y_1/b_1, \ldots, y_n/b_n, x_1/f_{x_1}(y_1, \ldots, y_n), \ldots, x_m/f_{x_m}(y_1, \ldots, y_n)]}(G) = 1$. As $\forall b_1, \ldots, b_n \in D_\nu$ it occurs that $f^\nu_{x_i}(b_1, \ldots, b_n) = a_i \in D_\nu$ and $\nu_{[y_1/b_1, \ldots, y_n/b_n, x_1/a_1, \ldots, x_m/a_m]}(G) = 1$. Therefore, $\nu(F) = 1$.

$\Rightarrow$) Assume $\nu(F) = 1$. Then, for all $\forall b_1, \ldots, b_n \in D_\nu$, there exists $a_1, \ldots, a_n \in D_\nu$ such that $\nu_{[y_1/b_1, \ldots, y_n/b_n, x_1/a_1, \ldots, x_m/a_m]}(G) = 1$. Since ν does not define f_{x_i} because they were taken to be a new function that did not occur in G, if ν' is an extension of ν that defines f_{x_i} such that $f^\nu_{x_i}(b_1, \ldots, b_n) = a_i$ then $\nu'_{[y_1/b_1, \ldots, y_n/b_n, x_1/f^\nu_{x_1}(b_1, \ldots, b_n), \ldots, x_m/f^\nu_{x_m}(b_1, \ldots, b_n)]}(G) = 1$, and thus

$$\nu'_{[y_1/b_1, \ldots, y_n/b_n]}(G[y_1/b_1, \ldots, y_n/b_n, x_1/f^\nu_{x_1}(b_1, \ldots, b_n), \ldots, x_m/f^\nu_{x_m}(b_1, \ldots, b_n)]) = 1.$$

Therefore, $\nu'(F') = 1$. □

We denote by $\mathcal{S}_k(F)$ the set of all the k-ary functions in the Skolem form of F. Thus, the set of constants is $\mathcal{S}_0(F)$.

3 Complexity of satisfiability

Given a logic $\mathcal{L}$ and a formula of that logic, the **decision problem** tries to answer whether that formula is valid or equivalently if it is satisfiable. Then, a **decision procedure** for the logic $\mathcal{L}$ is an algorithm that solves the decision problem for any formula in $\mathcal{L}$. We will say that a decision procedure is **sound** if it always returns the correct answer, and it is **complete** if it answers in a finite time. A **decidable** logic is one for which there exists a sound and complete decision procedure.

In this section, we will see the Herbrand expansion theorem, which is used along with the tiling problem, to prove that first order logic is undecidable and the satisfiability problem in the propositional logic is NP-complete. We begin by defining the *Herbrand universe* inductively.

If $D_0 = \begin{cases} \mathcal{S}_0(F) & \text{if } \mathcal{S}_0(F) \neq \varnothing \\ \{\tilde{1}\} & \text{otherwise, where } \tilde{1} \text{ is some arbitrary constant} \end{cases}$

and $D_n = \{f(t_1, \ldots, t_n) : f \in \mathcal{S}_n(F),\ t_1, \ldots, t_n \in \bigcup_{i=0}^{n-1} D_i\}$.

The **Herbrand universe** is $D(F) = \bigcup_{n\in\mathbb{N}} D_n$.

Example. 3.1.
We take F and v from the Example (2.2.) where we can deduce that F is satisfiable. The prenex form is $\forall y \forall w \exists x \exists z.(y < x) \wedge \neg(w < z)$. To remove $\exists x \exists z$, we choose the function $f(y) = y + 1$ and the constant $a = 0$ so that the subformulas $y < f(y)$ and $\neg(w < a)$ are always true.

Then, the *Herbrand universe* of F is the set of terms that can be obtained from a and f:

$$D(F) = \{a, f(a), f(f(a)), \ldots\}.$$

The result of removing the quantifiers of a formula F is called **matrix** of F, and it is denoted by F^*. We define the *Herbrand expansion* as the set of all formulas resulting from substituting the terms $D(F)$ in the matrix F^*.

Let $F = \forall y_1 \cdots \forall y_n F^*$ be a closed formula in Skolem form, where F^* is the matrix of F. The **Herbrand expansion** of F is the set of formulas

$$E(F) = \{F^*[y_1/t_1, \ldots, y_n/t_n] : t_1, \ldots, t_n \in D(F)\}.$$

Every formula in the *Herbrand expansion* can be treated as a proposition, i.e., $E(F) \subset \mathcal{L}_P$.

Example. 3.2.
Continuing the Example (3.1.), the matrix of F is $F^* = (y < f(y)) \wedge \neg(w < a)$ and the Herbrand expansion is

$$\begin{aligned} E(F) = \{(a < f(a)) \wedge \neg(a < a) &= F^*[y/a, w/a], \\ (f(a) < f(f(a))) \wedge \neg(a < a) &= F^*[y/f(a), w/a], \\ (a < f(a)) \wedge \neg(f(a) < a) &= F^*[y/a, w/f(a)], \\ (f(a) < f(f(a))) \wedge \neg(f(a) < a) &= F^*[y/f(a), w/f(a)], \ldots\}. \end{aligned}$$

Given a formula F in Skolem form, a **Herbrand interpretation** $\tau = (D_\tau, \alpha_\tau)$ satisfies:

- $D_\tau = D(F)$,
- for every function symbol $f \in \mathcal{S}_n(F)$ and $t_1, \ldots, t_n \in D(F)$, $f^\tau(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$.

By definition, the interpretation of function symbols and the domain of *Herbrand interpretations* is fixed.

In this result, it is proven that the use of Herbrand interpretations to determine if a formula is satisfiable is sufficient.

Lemma. 3.3.
A closed formula F in Skolem form has a model ν if and only if it has a Herbrand model τ.

PROOF.

$\Leftarrow$) If F has a Herbrand model then it has a model.

$\Rightarrow$) We define $\mu : D(F) \to D_\nu$ as the function that

- maps $\tilde{1} \to \mu(\tilde{1})$, where $\tilde{1}$ is an arbitrary constant in case F does not have any constant and $\mu(\tilde{1}) \in D_\nu$,
- and for each n-ary function $f \in \mathcal{S}_n(F)$ and $t_1, \ldots, t_n \in D(F)$, $\mu(f(t_1, \ldots, t_n)) = f^\nu(\mu(t_1), \ldots, \mu(t_n))$.

It maps the elements of $D(F)$ to elements of D_ν. The interpretation of the predicates is also taken from the interpretation ν:

$$(t_1, \ldots, t_k) \in P^\tau \Longleftrightarrow (\mu(t_1), \ldots, \mu(t_k)) \in P^\nu.$$

Now, we can observe that τ is somehow a restriction of ν to $D(F)$. It is a valid domain and $\mu(D(F)) \subseteq D_\nu$ then, τ is also a model of F.

As $\nu(F) = \nu(\forall y_1 \cdots \forall y_n G) = 1$, then for all $a_1, \ldots, a_n \in D_\nu$, $\nu_{[y_1/a_1, \ldots, y_n/a_n]}(G) = 1$. Since $\mu(D(F)) \subseteq D_\nu$, for all $t_1, \ldots, t_n \in D(F)$ occurs $\nu_{[y_1/\mu(t_1), \ldots, y_n/\mu(t_n)]}(G) = 1$. As a result, by the definition of μ and τ, for all $t_1, \ldots, t_n \in D(F)$ $\tau_{[y_1/t_1, \ldots, y_n/t_n]}(G) = 1$. Therefore, $\tau(F) = \tau(\forall y_1 \cdots \forall y_n G) = 1$. □

Theorem. 3.4. (Expansion theorem)
A closed formula is satisfiable if and only if its Herbrand expansion is satisfiable.

PROOF. The Lemma (2.8.) let us compute the Skolem form of $F = \forall y_1 \cdots \forall y_n G$ and it is equisatisfiable. By Lemma (3.3.) F is satisfiable if and only if there exist a Herbrand model. It only remains to prove that F has a Herbrand model if and only if its Herbrand expansion $E(F)$ is satisfiable.

$\Rightarrow$) Assume that τ is a Herbrand model of F, then for all $t_1, \ldots, t_n \in D(F)$, taking the interpretation $\tau_{[y_1/t_1, \ldots, y_n/t_n]}$ that evaluates y_i as t_i, we know that $\tau_{[y_1/t_1, \ldots, y_n/t_n]}(G) = 1$. Applying $\tau_{[y_1/t_1, \ldots, y_n/t_n]}$, then $\tau(G[y_1/t_1, \ldots, y_n/t_n]) = 1$ for all $t_1, \ldots, t_n \in D(F)$. Associating G with the matrix of F and bearing in mind that τ is a Herbrand interpretation, we obtain that for every $H \in E(F)$ occurs that $\tau(H) = 1$. Therefore, τ is a model of $E(F)$.

$\Leftarrow$) Assume that τ is a model of $E(F)$. Then, for all $H \in E(F)$ occurs that $\tau(H) = 1$. As $f(t_1, \ldots, t_n) = f^\tau(t_1, \ldots, t_n)$ for all $f \in \mathcal{S}_n(F)$ and all $t_1, \ldots, t_n \in D(F)$, since τ is a Herbrand interpretation, then for all $t_1, \ldots, t_n \in D(F)$, we have $\tau(G[y_1/t_1, \ldots, y_n/t_n]) = 1$. This is

equivalent to, for all $t_1, \ldots, t_n \in D(F)$ occurs $\tau_{[y_1/t_1,\ldots,y_n/t_n]}(G) = 1$. Finally, we conclude that τ is a Herbrand model of F.

□

To prove the undecidability of first order logic and that the satisfiability problem in PL is NP-complete, we introduce the tiling problem. It allows to abstract from computer theory. We are given a finite set of tiles D. The aim is to tile the first quadrant of the plane, as we can see in the image I.2. Given the tiling system $\mathcal{D} = (D, d_1, H, V)$ a tiling $\mathcal{T} : \mathbb{N} \times \mathbb{N} \to D$ by system $\mathcal{D}$ must satisfy:

1. $\mathcal{T}(1,1) = d_1$,
2. $(\mathcal{T}(m,n), \mathcal{T}(m+1,n)) \in H$,
3. $(\mathcal{T}(m,n), \mathcal{T}(m,n+1)) \in V$.

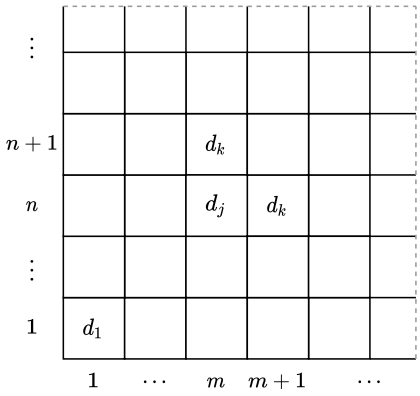


Figure I.2: Tiling problem

In the tiling system, D is the set of tiles with which the space is tiled, d_1 is the initial tile that is located at $(1,1)$, and H, V specify which pairs of tiles can be next to each other in the horizontal or vertical direction respectively. A tiling $\mathcal{T}$ is a function that assigns a tile to every position.

The tiling problem is undecidable, and the proof can be found in [32]. This result can be used to prove the next theorem.

Theorem. 3.5. (Church)
The satisfiability problem in first order logic is undecidable.

PROOF. The idea of the proof is to reduce the tiling problem to the satisfiability problem. Thus, since the tiling problem is undecidable, we conclude that the satisfiability problem for first order logic is also undecidable. For this purpose, we give a process to construct a formula $F_{\mathcal{D}}$ from a tiling system $\mathcal{D}$ so that there is a tiling for the tiling system $\mathcal{D}$ if and only if $F_{\mathcal{D}}$ is satisfiable.

Given a tiling system $\mathcal{D} = (D, d_1, H, V)$, we can construct the formula

$$F_{\mathcal{D}} = \forall x \forall y \Bigg(\bigwedge_{\substack{d_j, d_k \in D \\ j<k}} \neg(P_{d_j}(x,y) \wedge P_{d_k}(x,y)) \tag{I.1}$$

$$\wedge P_{d_1}(a,a) \tag{I.2}$$

$$\wedge \Bigg(\bigvee_{(d_j,d_k)\in H} (P_{d_j}(x,y) \wedge P_{d_k}(f(x),y)) \Bigg) \tag{I.3}$$

$$\wedge \Bigg(\bigvee_{(d_j,d_k)\in V} (P_{d_j}(x,y) \wedge P_{d_k}(x,f(y))) \Bigg) \Bigg), \tag{I.4}$$

where $P_d(m,n)$ is meant to be true when a tile $d \in D$ is assigned to the position (m,n) of the plane.

$\Rightarrow$) Suppose that there is a tiling $\mathcal{T} : \mathbb{N} \times \mathbb{N} \to D$ by $\mathcal{D}$. It is a tiling such that

$$\begin{array}{ll} \mathcal{T}(1,1) = d_1 & \\ (\mathcal{T}(m,n), \mathcal{T}(m+1,n)) \in H & \text{for all } m,n \in \mathbb{N} \\ (\mathcal{T}(m,n), \mathcal{T}(m,n+1)) \in V & \text{for all } m,n \in \mathbb{N}. \end{array}$$

Then the interpretation ν defined from $\mathcal{T}$

$$\begin{array}{ll} D\nu = \mathbb{N} & \\ a^{\nu} = 1 & \\ P_d^{\nu} = \{(m,n) : \mathcal{T}(m,n) = d\} & \text{for all } d \in D \\ f^{\nu}(z) = z+1 & \text{for all } z \in \mathbb{N}. \end{array}$$

is a model of $F_{\mathcal{D}}$, since

1. for every $m,n \in \mathbb{N}$ and every $d_j, d_k \in D$ such that $d_j \neq d_k$, it does not occur that $\mathcal{T}(m,n) = d_j$ and $\mathcal{T}(m,n) = d_k$, then the subformula I.1 is satisfied;

2. as $a^{\nu} = 1$ and $\mathcal{T}(1,1) = d_1$, then the subformula I.2 is satisfied;

3. for all $m,n \in \mathbb{N}$ there is $(d_j, d_k) \in H$ (resp. V) such that $\mathcal{T}(m,n) = d_j$ and $\mathcal{T}(m+1,n) = d_k$ (resp. $\mathcal{T}(m,n+1) = d_k$), then the subformula I.3 (resp. I.4) is satisfied;

$\Leftarrow$) Suppose that $F_{\mathcal{D}}$ is satisfiable, then there is a Herbrand model τ such that $D_{\tau} = D(F_{\mathcal{D}}) = \{a, f(a), f(f(a)), \ldots, f^n(a), \ldots\}$. We define the tiling $\mathcal{T} : \mathbb{N} \times \mathbb{N} \to D$ as

$$\mathcal{T}(m,n) = d \text{ if and only if } \tau(P_d(f^m(a), f^n(a))) = 1.$$

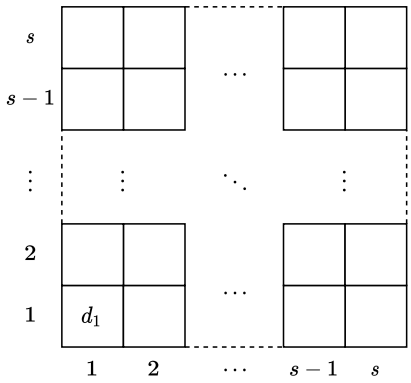
The tiling $\mathcal{T}$ is well-defined since for all $m,n \in \mathbb{N}$ the subformula

$$\neg(P_{d_j}(f^m(a), f^n(a)) \wedge P_{d_k}(f^m(a), f^n(a)))$$

assures that the position $(m,n) \in \mathbb{N} \times \mathbb{N}$ is not assigned to both d_j and d_k. Every position $(n,m) \in \mathbb{N} \times \mathbb{N}$ is assigned a tile d from D since for all $f^m(a), f^n(a) \in D(F_{\mathcal{D}})$ the subformula I.3 requires that $\nu(P_d(f^m(a), f^n(a))) = 1$ for some $d \in D$. From subformula I.1 is deduced that $\mathcal{T}(1,1) = d_1$.

To check that the horizontal constraints are satisfied, we observe that for every $f^m(a), f^n(a) \in D(F_{\mathcal{D}})$, the subformula I.3 is `true` if there exists $(d_j, d_k) \in H$ such that $\tau(P_{d_j}(f^m(a), f^n(a)) \wedge P_{d_k}(f^{m+1}(a), f^n(a))) = 1$. Therefore, by the definition of $\mathcal{T}$ we conclude that $(\mathcal{T}(m,n), \mathcal{T}(m+1,n)) \in H$ for all $m,n \in \mathbb{N}$. The vertical constraints are checked analogously. □

To prove that the satisfiability problem on the propositional logic is NP-complete, we use the bounded tiling problem. Given a tiling system $\mathcal{D} = (D, d_1, H, V)$, it is nothing more than to find a tiling $\mathcal{T} : \{1, \ldots, s\} \times \{1, \ldots, s\} \to D$ by $\mathcal{D}$ for the region $s \times s$.



The bounded tiling problem is NP-complete. The proof can be found in [32].

Theorem. 3.6. (Cook)
The satisfiability problem in propositional logic is NP-complete.

PROOF. The satisfiability problem in propositional logic is in NP, since we can verify a satisfying assignment in polynomials time.

To prove that it is NP-complete, we reduce the bounded tiling problem, which is NP-complete, to the satisfiability problem in propositional logic. To do so, the formula $P_{\mathcal{D},s}$ is defined as follows:

1. Compute $F_{\mathcal{D}}$ as it was done in the proof of Church's Theorem (3.5.)
2. Only consider the elements of $E^s(F_{\mathcal{D}}) \subset E(F_{\mathcal{D}})$, where
$$E^s(F_{\mathcal{D}}) = \{F^*_{\mathcal{D}}[x/f^j(a), y/f^k(a)] : 1 \leqslant j,k \leqslant s\}.$$
3. As it can be observed, $E^s(F_{\mathcal{D}})$ is a finite set. Thus, $P_{\mathcal{D},s} = \bigwedge_{P\in E^s(F_{\mathcal{D}})} P$ is the conjunction of all the formulas in the bounded Herbrand expansion. It can be seen as a formula of the propositional logic.

So defined, $P_{\mathcal{D},s}$ is satisfiable if and only if there exists a tiling for the bounded tiling problem. □

4 First order theories

First order theories formalize structures like numbers, lists and arrays, by the definition of an appropriate set of axioms that enables reasoning about them. A first order **theory** T is defined by

1. a **signature** Σ: set of constant, function and predicate symbols,
2. a set of **axioms**: set of closed FOL formulas whose constant, function, and predicate symbols are taken from Σ.

A Σ**-formula** is a FOL formula whose constant, function and predicate symbols belong to Σ. An interpretation is a model of a set of axioms, if the interpretation is a model for each of the axioms. A Σ-formula is **T-valid** if every model of the set of axioms is also a model of the Σ-formula. A theory consists of all T-valid Σ-formulas.

Theory of equality

The signature Σ_E of the **theory of equality** T_E, consist of a binary predicate that is meant to be the equality $=$, and all constant, function, and predicate symbols. The meaning of the predicate symbol $=$ is defined by the axioms of T_E:

1. $\forall x.x = x$ (reflexivity)
2. $\forall x, y.x = y \rightarrow y = x$ (symmetry)
3. $\forall x, y, z.x = y \wedge y = z \rightarrow x = z$ (transitivity)
4. for any n-ary function symbol f, (function congruence)

   $$\forall x_1, \ldots, x_n, y_1, \ldots, y_n. \left(\bigwedge_{i=1}^{n} x_i = y_i \right) \rightarrow f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n)$$

5. for any n-ary predicate symbol P, (predicate congruence)

   $$\forall x_1, \ldots, x_n, y_1, \ldots, y_n. \left(\bigwedge_{i=1}^{n} x_i = y_i \right) \rightarrow P(x_1, \ldots, x_n) = P(y_1, \ldots, y_n).$$

It is easy to observe that the formula $x = y \wedge y = z \rightarrow g(f(x), y) = g(f(z), x)$ is an example of a formula that is T_E-valid, since all the models of the axioms are also models of this Σ_E-formula.

Theory of Presburger Arithmetic

The signature $\Sigma_{\mathbb{N}} = \{0, 1, +, =\}$ of the theory of Presburger arithmetic $T_{\mathbb{N}}$ consist of the constants 0 and 1, the binary function $+$ and the binary predicate $=$. Its axioms are:

1. $\forall x.\neg(x + 1 = 0)$ (zero)
2. $\forall x, y.x + 1 = y + 1 \rightarrow x = y$ (successor)
3. $F(0) \wedge (\forall x.F(x) \rightarrow F(x + 1)) \rightarrow \forall x.F(x)$ (induction)
4. $\forall x.x + 0 = x$ (plus zero)
5. $\forall x, y.x + (y + 1) = (x + y) + 1$ (plus successor)

where in 3, F is a $\Sigma_{\mathbb{N}}$-formula.

Theory of real closed fields

The **theory of reals** $T_{\mathbb{R}}$ has signature $\Sigma_{\mathbb{R}} = \{0, 1, +, -, \cdot, =, \geqslant\}$ where $0, 1$ are constants. Its axioms are:

1. $\forall x, y.x \geqslant y \wedge y \geqslant x \rightarrow x = y$ (antisimetry)
2. $\forall x, y, z.x \geqslant y \wedge y \geqslant z \rightarrow x \geqslant z$ (transitivity)
3. $\forall x, y.x \geqslant y \vee y \geqslant x$ (totality)

4. $\forall x, y, z.(x + y) + z = x + (y + z)$ (+ associativity)
5. $\forall x.x + 0 = x$ (+ identity)
6. $\forall x.x + (-x) = 0$ (+ inverse)
7. $\forall x, y.x + y = y + x$ (+ commutativity)
8. $\forall x, y, z.x \geqslant y \rightarrow x + z \geqslant y + z$ (+ ordered)

9. $\forall x, y, z.(x \cdot y) \cdot z = x \cdot (y \cdot z)$ ($\cdot$ associativity)
10. $\forall x.x \cdot 1 = x$ ($\cdot$ identity)
11. $\forall x.\neg(x = 0) \rightarrow \exists y.x \cdot y = 1$ ($\cdot$ inverse)
12. $\forall x, y.x \cdot y = y \cdot x$ ($\cdot$ commutativity)
13. $\forall x, y.x \geqslant 0 \wedge y \geqslant 0 \rightarrow x \cdot y \geqslant 0$ ($\cdot$ ordered)
14. $\forall x, y, z.x \cdot (y + z) = x \cdot y + x \cdot z$ (distributivity)
15. $\neg(0 = 1)$ (separate identities)
16. $\forall x \exists y.x = y^2 \vee -x = y^2$ (square-root)
17. for each odd integer n,
    $\forall x_1, \ldots, x_n \exists y.y^n + x_1 \cdot y^{n-1} + \cdots + x_{n-1} \cdot y + x_n = 0$ (at least one root)

From 1 to 3, $\geqslant$ is expected to be a **total order**. From 4 to 7, are the axioms of a commutative group under addition $+$. From 9 to 12, are the axioms of a commutative group under multiplication. In 15 is set that the additive and multiplicative identities are different. Finally, the additional axioms of a **real closed field** are 8, 13, 16 and 17.

Fragments

A **fragment** of a theory is a syntactically restricted subset of formulas of the theory. An example is the **quantifier-free fragment** of a theory T. It is the set of formulas of T without quantifiers. We assume the convention that non-closed formulas are valid if the universal closure is valid. Although satisfiability in FOL is undecidable, satisfiability in particular theories or fragments of theories is sometimes decidable, which allows the automation of these problems. The fragments of first order logic that are decidable are already known and classified. The classification can be found in [13].

Chapter II

Gröbner bases

We used *Gaussian elimination* on systems of linear equations to obtain a new set of linear equations with the same set of solutions, but which is easier to solve. For polynomials in one variable, the process analogous to Gaussian elimination is the *Euclidean algorithm*. In the process of calculating Gröbner bases, we obtain lower degree polynomials, but this time they are non-linear, multivariate polynomials.

We start by introducing the definition of a *variety* as the set of common zeros of a system of polynomial equations.

Let K be a field and $f_1, \ldots, f_s \in K[X_1, \ldots, X_n]$, the set

$$V(f_1, \ldots, f_s) = \{\overline{s} \in K^n; f_i(\overline{s}) = 0, \forall i = 1, \ldots, s\}$$

is called the **variety** of the system of polynomial equations $f_1, \ldots, f_s$.

In the problem of satisfiability checking, it is not needed to calculate the solutions of the system, but to check if there is any solution. This will be seen in the next chapter, and we will see how to check this in the Hilbert's Nullstellensatz Theorem (11.3.).

This chapter is based on [17, 28].

5 Ideals

To compute an equivalent system of equations, we need the notion of *ideal*.

A subset $I \subset K[X_1,\dots,X_n]$ is an **ideal** if it satisfies:

- $0 \in I$
- if $f, g \in I$, then $f + g \in I$
- if $f \in I$ and $h \in K[X_1,\dots,X_n]$. Then, $hf \in I$.

Given $f_1,\dots,f_s \in K[X_1,\dots,X_n]$, the **ideal generated by** $f_1,\dots,f_s$ is

$$\langle f_1,\dots,f_s\rangle = \left\{ \sum_{i=1}^{s} h_i f_i : h_i \in K[X_1,\dots,X_n] \right\}.$$

In the next result, we check that $\langle f_1,\dots,f_s\rangle$ is actually an ideal.

Lemma. 5.1.
Given $f_1,\dots,f_s \in K[X_1,\dots,X_n]$, then $\langle f_1,\dots,f_s\rangle$ is an ideal of $K[X_1,\dots,X_n]$.

PROOF.

- $0 \in \langle f_1,\dots,f_s\rangle$ since $\sum_{i=0}^{s} 0 \cdot f_i = 0$.

Suppose that $f = \sum_{i=1}^{s} p_i f_i$, $g = \sum_{i=1}^{s} q_i f_i$ and $h \in K[X_1,\dots,X_n]$. Then,

- $f + g = \sum_{i=1}^{s} (p_i + q_i) f_i$,
- $hf = \sum_{i=1}^{s} (hp_i) f_i$.

□

We say that an ideal I is *finitely generated* if there exist $f_1,\dots,f_s \in K[X_1,\dots,X_n]$ such that $I = \langle f_1,\dots,f_s\rangle$, and $\{f_1,\dots,f_s\}$ is a **basis** of I.

Lemma. 5.2.
Let $f_1,\dots,f_s \in K[X_1,\dots,X_n]$ and $I = \langle f_1,\dots,f_s\rangle$, then $V(I) = V(f_1,\dots,f_s)$.

PROOF.

$\subset$) Consider $\bar{s} \in V(I)$, then $f(\bar{s}) = 0$ for all polynomials in I. As $f_1,\dots,f_s \in I$, we conclude that $\bar{s} \in V(f_1,\dots,f_s)$.

$\supset$) Let $\bar{s} \in V(f_1,\dots,f_s)$, then $f_1(\bar{s}) = \cdots f_s(\bar{s}) = 0$. As for all $f \in I$,

$$f(\bar{s}) = \sum_{i=1}^{s} h_i f_i(\bar{s}) = \sum_{i=1}^{s} h_i \cdot 0 = 0,$$

we conclude that $\bar{s} \in V(I)$. □

By definition of ideals generated by a set of polynomials, these ideals are the connecting point of systems of equations with the same set of solutions. An *ideal* determines a variety, and the different sets of generating equations of that ideal are different ways of expressing systems of equations that have the same common zeros. The purpose is to find a set of generating equations of an ideal such that the associate system of equations is easier to solve.

Theorem. 5.3.
Let $\{f_1,\dots,f_s\}$ and $\{g_1,\dots,g_t\}$ be bases of the same ideal, so that $\langle f_1,\dots,f_s\rangle = \langle g_1,\dots,g_t\rangle$. Then, $V(f_1,\dots,f_s) = V(g_1,\dots,g_t)$.

PROOF. Given an ideal I, we start by probing that $V(f_1,\dots,f_s) = V(I)$ for any basis $\{f_1,\dots,f_s\}$ of I.

$\subset$) Given $\bar{s} \in V(f_1, \ldots, f_s)$, since

$$I = \langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i : h_i \in K[X_1, \ldots, X_n] \right\},$$

then for any $f \in I$ we have $f(\bar{s}) = \sum_{i=1}^{s} h_i(\bar{s}) \cdot 0 = 0$. Therefore, $\bar{s} \in V(I)$.

$\supset$) Given $\bar{s} \in V(I)$, since $f_1, \ldots, f_s \in I$, then $f_i(\bar{s}) = 0$ for all i. Therefore, $\bar{s} \in V(f_1, \ldots, f_s)$.

Finally, as $\{f_1, \ldots, f_s\}$ and $\{g_1, \ldots, g_t\}$ are bases of I, then we have the equality $V(f_1, \ldots, f_s) = V(I) = V(g_1, \ldots, g_t)$. □

6 Monomial orderings

We consider a relationship between monomials and its exponents as tuples of natural numbers,

$$X^{\alpha} = X_1^{\alpha_1}, \ldots, X_n^{\alpha_n} \longleftrightarrow \alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n.$$

It is also set the order of the variables, which is their location within the tuple, the usual case is $X_1 > \cdots > X_n$.

To operate on non-linear, multivariate polynomials, we want to find an order to arrange the monomials of a polynomial by their exponents. But it would be desired that this ordering allows us to compute the division of polynomials similarly to how we used to do it for linear equations or polynomials in one variable. Thus, we define a *monomial ordering* as follows.

A **monomial ordering** is an ordering $\preceq$ on $\mathbb{N}^n$, which is:

- **total**,
- **compatible**: if $\alpha \preceq \beta$ and $\gamma \in \mathbb{N}^n$, then $\alpha + \gamma \preceq \beta + \gamma$,
- **admissible**: $(0, \ldots, 0)$ is the minimum of $\mathbb{N}^n$.

From the first condition, we know that only one of the following is applied:

$$X^{\alpha} > X^{\beta} \quad X^{\alpha} = X^{\beta} \quad X^{\alpha} < X^{\beta}.$$

The second condition assures that when multiplying a polynomial by a monomial, the order of the monomials does not change. As it is a total and admissible ordering, it is a **well-ordering**. This is, every non-empty subset of $\mathbb{N}^n$ has a minimum. Therefore, any strictly decreasing sequence has an end, this is necessary so that the algorithms defined later finish.

From the definition above, we can deduce that monomial orderings also fulfil the next property.

Corollary. 6.1.
Let $\preceq$ be a compatible and admissible ordering. Given $\alpha, \beta \in \mathbb{N}^n$ and $\alpha_i < \beta_i$ for every i, then $\alpha \preceq \beta$.

PROOF. If $\alpha_i \leqslant \beta_i$ for every i, there exists $\gamma \in \mathbb{N}^n$ such that $\alpha + \gamma = \beta$. As $0 \preceq \gamma$, then $\alpha \preceq \alpha + \gamma = \beta$. □

Although there are more monomial orderings such as *graded lex* and *graded reverse lex order*, here we will only define the *lexicographic order* and assume that it is the one used from now on.

Given $\alpha, \beta \in \mathbb{N}^n$, we say $\alpha \succ_{lex} \beta$ if the left-most non-zero entry of $\alpha - \beta \in \mathbb{Z}^n$ is positive. The **lexicographic order** is defined by $\succ_{lex}$.

In the image below, we can get an insight about how the monomial exponents are arranged in a lexicographic order.

There are many monomial orders, and once a monomial order is fixed, there are $n!$ orderings corresponding to how the n variables are ordered. Normally, computer algebra systems are programmed to support lex, graded lex and graded reverse orders and to choose the priority order of the variables.

Given $0 \neq f = \sum_\alpha a_\alpha X^\alpha \in K[X_1, \ldots, X_n]$ and a certain monomial order, we define the following concepts:

- **exponent**: $\exp(f) = max(\alpha \in \mathbb{N}^n : a_\alpha \neq 0)$.

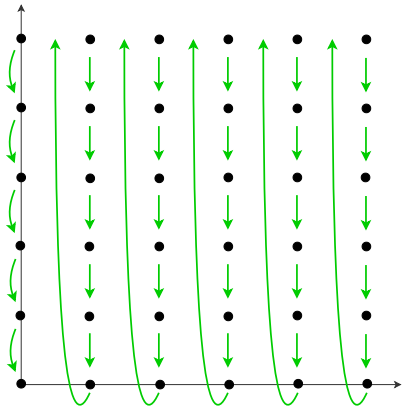


Figure II.1: Representation of lexicographic order for 2 variables.

- **leading coefficient**: $LC(f) = a_{\exp(f)} \in K$.
- **leading monomial**: $LM(f) = x^{\exp(f)}$.
- **leading term**: $LT(f) = LC(f) \cdot LM(f)$.

If $f = 0 \in K[X_1, \ldots, X_n]$, then $LC(f) = LT(f) = 0$.

The exponent has the following properties:

Lemma. 6.2.
Given $0 \neq f, g \in K[X_1, \ldots, X_n]$, *then:*

- $exp(f\,g) = exp(f) + exp(g)$,
- *if* $f + g \neq 0$, *then* $exp(f + g) \preceq max(exp(f), exp(g))$,
- *if* $exp(f) \prec exp(g)$, *then* $exp(f + g) = exp(g)$.

7 Division algorithm

Given the polynomial $f \in K[X_1,\dots,X_n]$, we would like to express it in terms of $f_1,\dots,f_s \in K[X_1,\dots,X_n]$, as it will help us to decide if $f \in \langle f_1,\dots,f_s\rangle$.

Theorem. 7.1.
Fixed a monomial order $\succ$, and let $F = \{f_1,\dots,f_s\} \subset K[X_1,\dots,X_n]$ be an ordered set of polynomials. Then, every $f \in K[X_1,\dots,X_n]$ can be written as

$$f = a_1 f_1 + \cdots + a_s f_s + r,$$

where $a_i, r \in K[X_1,\dots,X_n]$, and either $r = 0$ or r is a linear combination, with coefficients in K, of monomials, none of which is divisible by any of $LT(f_1),\dots,LT(f_s)$. We call r a remainder of f on division by F. Furthermore, if $a_i f_i \neq 0$, then we have $exp(f) \succeq exp(a_i f_i)$.

To get such expression, we define the division algorithm for non-linear multivariate polynomials. Due to the *monomial ordering* definition, the result of dividing one polynomial by another gives us a polynomial whose leading term is less than that of the starting polynomial, and it is assured that the division algorithm always ends.

DIVISION_ALGORITHM$(f, f_1,\dots,f_s)$**:**

Input: $f, f_1,\dots,f_s \in K[X_1,\dots,X_n]$

Result: $a_1,\dots,a_s, r$ such that $f = \sum_{i=1}^{s} a_i f_i + r$

$a_1 = \cdots = a_s = r = 0$

$p = f$

while $p \neq 0$:

if $\{i; LT(f_i)$ divides $LT(P)\} \neq \varnothing$:

$i = min\{i; LT(f_i)$ divides $LT(P)\}$

$a_i = a_i + \frac{LT(p)}{LT(f_i)}$

$p = p - \frac{LT(p)}{LT(f_i)} f_i$

else:

$r = r + LT(p)$

$p = p - LT(p)$

It is clear that when this algorithm returns remainder equals 0, in the division of f by the set $G = \{f_1, \ldots, f_s\}$, then f can be express as a combination of elements of G. Therefore, f belongs to the ideal I generated by G. This is the so-called *Ideal Membership problem*. We will see that for $f \in I$, it might happen that the remainder returned by the *division algorithm* is not 0 for any basis of the ideal, but there exists a kind of bases that allows as to solve the *Ideal Membership problem*. These bases are called Gröbner bases.

In the next example, we see that if we change the order of the divisor polynomials, the remainder might be different.

Example. 7.2.
Given the set of polynomials $G = \{f_1 = XY + 1, f_2 = Y^2 + 1\}$ as a generator of the ideal $I = \langle f_1, f_2 \rangle$ and $f = (XY + 1)^3 = X^3Y^3 + 3X^2Y^2 + 3XY + 1 \in I$, we want to use the division algorithm to divide f by G.

First, we will consider that the polynomials in G are ordered as shown before, i.e., $f_1 = XY + 1, f_2 = Y^2 + 1$. As a result, we get $f = (X^2Y^2 + 2XY + 1) \cdot f_1$.

	p	$XY+1$	Y^2+1	r
0	$X^3Y^3 + 3X^2Y^2 + 3XY + 1$	X^2Y^2	0	0
1	$2X^2Y^2 + 3XY + 1$	$2XY$	0	0
2	$XY + 1$	1	0	0

Now, changing the order of the set of polynomials by which we divide, e.i., $f_1 = Y^2 + 1, f_2 = XY + 1$, we get a different result $f = (X^3Y + 3X^2) \cdot f_1 + (-X^2 + 3) \cdot f_2 - 2X^2 - 2$.

	p	Y^2+1	$XY+1$	r
0	$X^3Y^3+3X^2Y^2+3XY+1$	X^3Y	0	0
1	$-X^3Y+3X^2Y^2+3XY+1$	0	$-X^2$	0
2	$3X^2Y^2+X^2+3XY+1$	$3X^2$	0	0
3	$-2X^2+3XY+1$	0	0	$-2X^2$
4	$3XY+1$	0	3	0
5	-2	0	0	-2

Comparing both results, we observe that not only the remainder is not the same, but the coefficients are also different. We can also observe that

$$r = -2X^2 - 2 = -2((Xf_2 - Yf_1)^2 - f_2 + 2f_1) \in I,$$

but it is not divisible by any of the polynomials of the basis. In the next section we define the *S*–polynomials; they will be useful to make sure that the remainder is not in the ideal.

8 S-polynomials

There may exist polynomials in I that are not divisible by the elements of a basis, but they can be divided by polynomials that are a combination of polynomials in I involving the cancellation of leading terms.

Example. 8.1.
We take the polynomial $f = Y \cdot f_1 - X \cdot f_2 = -X + Y \in I$, where $f_1 = XY + 1, f_2 = Y^2 + 1$ and $I = \langle f_1, f_2 \rangle$. It turns out, that f is not divisible by any of the polynomials of the basis $G = \{f_1, f_2\}$.

A polynomial $f \in I = \langle f_1, \ldots, f_s \rangle$ for which the division algorithm does not return zero when dividing by $G = \{f_1, \ldots, f_s\}$, is a combination of polynomials of G, where leading terms are cancelled. Term cancellation allows the underlying monomials to appear as leading terms. *S-polynomials* are combinations of polynomials with cancellation of leading terms.

Given $0 \neq f, g \in K[X_1, \ldots, X_n]$.

- If $\exp(f) = \alpha$ and $\exp(g) = \beta$, then let $\gamma = (\gamma_1, \ldots, \gamma_n)$, where $\gamma_i = max(\alpha_i, \beta_i)$.
  The **least common multiple** of $LM(f)$ and $LM(g)$ is $X^\gamma = LCM(LM(f), LM(g))$.

- The **S-polynomial** of f and g is $S(f,g) = \frac{X^\gamma}{LT(f)} \cdot f - \frac{X^\gamma}{LT(g)} \cdot g$.

Ideally, we want that every polynomial of the ideal is divisible by the polynomials of the basis, so we can easily solve the ideal membership problem by just using the division algorithm. It is possible to find a basis of the ideal for which any polynomial in the ideal is divisible by the basis. It can be achieved with S-polynomials.

Lemma. 8.2.
Suppose we have $\sum_{i=1}^s c_i f_i$, where $c_i \in K$ and $exp(f_i) = \delta \in \mathbb{N}^n$ for all i. If $exp(\sum_{i=1}^s c_i f_i) \prec \delta$, then $\sum_{i=1}^s c_i f_i$ is a linear combination, with coefficients in K, of the S-polynomials $S(f_j, f_k)$ for $1 \leqslant j < k \leqslant s$. Furthermore, each $S(f_j, f_k)$ has an exponent $< \delta$.

PROOF. Denoting $d_i = LC(f_i)$, the leading coefficient of $c_i f_i$ is $c_i d_i$. Since $LT(c_i f_i) = c_i d_i x^\delta$ have exponent δ and $\exp\left(\sum_{i=1}^s c_i f_i\right) \prec \delta$, we deduce that $\sum_{i=1}^s c_i d_i = 0$.

We are interested in $p_i = f_i/d_i$ because $LC(p_i) = 1$. Furthermore, consider

$$\sum_{i=1}^s c_i f_i = \sum_{i=1}^s c_i d_i p_i = c_1 d_1 (p_1 - p_2) + (c_1 d_1 + c_2 d_2)(p_2 - p_3) + \cdots + (c_1 d_1 + \cdots + c_{s-1} d_{s-1})(p_{s-1} - p_s) + (c_1 d_1 + \cdots + c_s d_s) p_s$$

As $LT(f_i) = d_i x^\delta$, then $x^\delta = LCM(LM(f_j), LM(f_k))$. Therefore,

$$S(f_j, f_k) = \frac{x^\delta}{LT(f_j)} f_j - \frac{x^\delta}{LT(f_k)} f_k = \frac{x^\delta}{d_j x^\delta} f_j - \frac{x^\delta}{d_k x^\delta} f_k = p_j - p_k. \tag{II.1}$$

Using equation II.1 and the fact that $\sum_{i=1}^s c_i d_i = 0$, we get as a result

$$\sum_{i=1}^s c_i f_i = c_1 d_1 S(f_1, f_2) + (c_1 d_1 + c_2 d_2) S(f_2, f_3) + \cdots + (c_1 d_1 + \cdots + c_{s-1} d_{s-1}) S(f_{s-1}, f_s)$$

that is a linear combination of S-polynomials. As $LT(p_j) = LT(p_k) = x^\delta$, then $\exp(p_j - p_k) \prec \delta$. By equation II.1, $\exp(S(f_j, f_k)) \prec \delta$.

□

9 Definition and properties

Let $I \subset K[X_1,\dots,X_n]$ be an ideal other than $\{0\}$ and

$$LT(I) = \{cX^\alpha : \exists f \in I \text{ with } LT(f) = cX^\alpha\},$$

we denote by $\langle LT(I)\rangle$ the ideal generated by the elements of $LT(I)$.

In our search for a basis of an ideal $I = \langle f_1,\dots,f_s\rangle$ for which the remainder on division of $f \in I$ by $\{f_1,\dots,f_s\}$ is always zero when using the division algorithm, we need a basis with the following property.

Fixed a monomial order. A finite subset $G = \{g_1,\dots,g_t\}$ of an ideal I is a **Gröbner basis** if

$$\langle LT(g_1),\dots,LT(g_t)\rangle = \langle LT(I)\rangle.$$

With this property, the leading term of polynomials in the ideal is a combination of the leading terms of the elements of the basis. Thus, every polynomial of the ideal is divisible by a polynomial of the basis. In this way, the division algorithm will always return zero when the polynomial is in the ideal.

Monomial ideals

For the next sort of ideal, we can determine that there is a finite set of elements of the ideal that generate it.

An ideal $I \subset K[X_1,\dots,X_n]$ is a **monomial ideal** if there is a subset $A \subset \mathbb{N}^n$ (possibly infinite) such that I consists of all polynomials which are finite sums of the form $\sum_{\alpha\in A} h_\alpha X^\alpha$, where $h_\alpha \in K[X_1,\dots,X_n]$. We write $I = \langle X^\alpha : \alpha \in A\rangle$.

This result proves that monomial ideals are finitely generated. But it is also used to prove that, for any ideal, there exists a Gröbner basis with a finite number of elements.

Lemma. 9.1. (Dickson's lemma)
A monomial ideal $I = \langle X^\alpha : \alpha \in A \rangle$ can be expressed as $I = \langle X^{\alpha(1)}, \ldots, X^{\alpha(s)} \rangle$, where $\alpha(1), \ldots, \alpha(s) \in A$ is a finite set of exponents.

The Hilbert basis theorem

Now, it is proven that for any ideal, there exists a Gröbner basis with a finite number of elements.

Theorem. 9.2.
For every ideal $I \subset K[X_1, \ldots, X_n]$, there exists a Gröbner basis of I.

PROOF. If $I = \{0\}$, its generating set is clearly finite.

For a non-zero ideal:

Existence of a finite number of $g_1, \ldots, g_t \in I$ such that $\langle LT(I) \rangle = \langle LT(g_1), \ldots, LT(g_t) \rangle$. As $\langle LT(I) \rangle$ is generated by the monomials $LM(g)$ where $g \in I - \{0\}$, it is a monomial ideal. Then, we can use Dickson's Lemma (9.1.) to deduce that $\langle LT(I) \rangle = \langle LM(g_1), \ldots, LM(g_t) \rangle$ for finitely many $g_1, \ldots, g_t \in I$. Since, $LT(g_1), \ldots, LT(g_t)$ differs from $LM(g_1), \ldots, LM(g_t)$ by a non-zero constant, $\langle LT(I) \rangle = \langle LT(g_1), \ldots, LT(g_t) \rangle$.

$\langle g_1, \ldots, g_t \rangle = I$, where $g_1, \ldots, g_t$ are the polynomials found in the previous paragraph (Hilbert basis theorem).

$\subset$) $\langle g_1, \ldots, g_t \rangle \subset I$ since $g_1, \ldots, g_t \in I$.

$\supset$) For $f \in I$, we apply the division algorithm to divide f by $\{g_1, \ldots, g_t\}$ to get

$$f = a_1 g_1 + \cdots + a_t g_t + r$$

where r is not divisible by any $LT(g_1), \ldots, LT(g_t)$. Then, $r = f - a_1 g_1 - \cdots - a_t g_t \in I$.

If $r \neq 0$, then $LT(r) \in \langle LT(I)\rangle = \langle LT(g_1),\dots,LT(g_t)\rangle$, so r is divisible by one of the following: $LT(g_1),\dots,LT(g_t)$, what is a contradiction. Therefore, $r = 0$ and $f \in \langle g_1,\dots,g_t\rangle$.

□

The remainder resulting from dividing a polynomial by a Gröbner basis is always the same, regardless of the order of the dividing polynomials.

Proposition. 9.3.
Let $G = \{g_1,\dots,g_s\}$ be a Gröbner basis for an ideal $I \subset K[X_1,\dots,X_n]$. Then, there is a unique $r \in K[X_1,\dots,X_n]$, such that no term of r is divisible by any of $LT(g_1),\dots,LT(g_t)$, no matter the order of the elements in G.

PROOF.

Existence. The division algorithm gives $f = a_1g_1 + \cdots + a_tg_t + r$ where r is not divisible by $LT(g_1),\dots,LT(g_s)$.

Uniqueness. Assume $f = a_1g_1 + \cdots + a_tg_t + r_1 = a_1^*g_1 + \cdots + a_t^*g_t + r_2$. Then $r_2 - r_1 = (a_1 - a_1^*)g_1 + \cdots + (a_t - a_t^*)g_t \in I$. If $r_1 \neq r_2$, then $LT(r_2 - r_1) \in \langle LT(I)\rangle = \langle LT(g_1),\dots,LT(g_t)\rangle$. This is impossible since no term of r_1, r_2 is divisible by any of $LT(g_1),\dots,LT(g_t)$. Therefore, $r_1 = r_2$. □

The **unique remainder** of f on division by G is denoted by $\mathcal{R}[f : G]$.

We can conclude that given a Gröbner basis $G = \{g_1,\dots,g_t\}$ of an ideal $I \subset K[X_1,\dots,X_n]$, we can determine whether a polynomial $f \in K[X_1,\dots,X_n]$ belongs to the ideal I by checking if the reminder on division of f by G is zero.

Corollary. 9.4.
Let $G = \{g_1,\dots,g_t\}$ be a Gröbner basis for an ideal $I \subset K[X_1,\dots,X_n]$ and let $f \in K[X_1,\dots,X_n]$. Then $f \in I$ if and only if the remainder on division of f by G is zero.

PROOF.

⇐) If the remainder is zero, it is trivial since $f = h_1g_1 + \cdots + h_tg_t + 0 \in I$, where $h_i \in K[X_1, \ldots, X_n]$.

⇒) If $f \in I$ then the division algorithm give us the expression $f = h_1g_1 + \cdots + h_tg_t + r$, where r is not divisible by any $LT(g_1), \ldots, LT(g_t)$ and by the Proposition (9.3.) it is unique. As $f = h_1g_1 + \cdots + h_tg_t + r \in I$ and $h_1g_1 + \cdots + h_tg_t \in I$, then $r \in I$. Suppose that $r \neq 0$, then $LT(r) \in \langle LT(I)\rangle = \langle LT(g_1), \ldots, LT(g_t)\rangle$. In this case, r is divisible by some $LT(g_1), \ldots, LT(g_t)$, what is a contradiction. Therefore, $r = 0$. □

10 Buchberger's algorithm

The Buchberger's algorithm computes the Gröbner basis of a set of multivariate polynomials, and the Buchberger's criterion ensures the correctness of the algorithm.

Theorem. 10.1. (Buchberger's criterion)
Let I be a polynomial ideal. Then, a basis $G = \{g_1, \ldots, g_t\}$ of I is a Gröbner basis if and only if $\mathcal{R}[S(g_i, g_j) : G] = 0$ for all pairs $i \neq j$, where the elements in G are listed in some order.

PROOF.

⇒) Since G is a Gröbner basis and $S(g_i, g_j) \in I$, the remainder on division by G is zero by Corollary (9.4.).

⇐) Given $0 \neq f \in I$, the aim is to show that if $\mathcal{R}[S(g_i, g_j) : G] = 0$ for all pairs $i \neq j$, then $LT(f) \in \langle LT(g_1), \ldots, LT(g_t)\rangle$, that is, f is divisible by some element of the basis.

Let $f \in I = \langle g_1, \ldots, g_t\rangle$, there are polynomials $h_i \in K[X_1, \ldots, X_n]$ such that

$$f = \sum_{i=1}^{t} h_i g_i. \tag{II.2}$$

We define $m(i) = \exp(h_i g_i)$ and $\delta = max(m(1), \ldots, m(t))$, then by Lemma (6.2.),

$$\exp(f) \preceq \delta.$$

Note that, if $\exp(f) = \delta$, then $\exp(f) = \exp(h_i g_i)$ for some i and $LT(f)$ is divisible by $LT(g_i)$, so $LT(f) \in \langle LT(g_1), \ldots, LT(g_t)\rangle$, what we want to prove. Considering all the possible ways that f can be written in the form (II.2), we observe that for some of those expressions we possibly get different δ. As a monomial order is a well-ordering, we take the minimal δ. Such a δ satisfies that $\exp(f) = \delta$, and we will prove that by contradiction.

Suppose that $\exp(f) \prec \delta$. The following form allows us to isolate the terms of f with exponent δ:

$$f = \sum_{m(i)=\delta} h_i g_i + \sum_{m(i)\prec\delta} h_i g_i = \sum_{m(i)=\delta} LT(h_i) g_i + \sum_{m(i)=\delta} (h_i - LT(h_i)) g_i + \sum_{m(i)\prec\delta} h_i g_i. \tag{II.3}$$

As the monomials in the second and third sums have exponent $\prec \delta$, we just need to check the assumption $\exp(f) \prec \delta$ in the first sum.

Denoting $LT(h_i) = c_i X^{\alpha(i)}$, we get $\sum_{m(i)=\delta} LT(h_i) g_i = \sum_{m(i)=\delta} c_i X^{\alpha(i)} g_i$ which has the form described in Lemma (8.2.), where $f_i = X^{\alpha(i)} g_i$. Therefore, by Lemma (8.2.), it is a linear combination of the S-polynomials $S(X^{\alpha(j)} g_j, X^{\alpha(k)} g_k)$, that we can also express as

$$\begin{aligned} S(X^{\alpha(j)} g_j, X^{\alpha(k)} g_k) &= \frac{X^\delta}{X^{\alpha(j)} LT(g_j)} X^{\alpha(j)} g_j - \frac{X^\delta}{X^{\alpha(k)} LT(g_k)} X^{\alpha(k)} g_k \\ &= \frac{X^\delta}{LT(g_j)} g_j - \frac{X^\delta}{LT(g_k)} g_k \\ &= X^{\delta - \gamma_{jk}} S(g_j, g_k), \end{aligned}$$

where $X^{\gamma_{jk}} = LMC(LM(g_j), LM(g_k))$. Thus, there are constants $c_{jk} \in K$ such that

$$\sum_{m(i)=\delta} LT(h_i) g_i = \sum_{m(i)=\delta} c_{jk} X^{\delta - \gamma_{jk}} S(g_j, g_k). \tag{II.4}$$

Now, we can use the hypothesis $\mathcal{R}[S(g_j, g_k) : G] = 0$ and the division algorithm to get the equality

$$S(g_j, g_k) = \sum_{i=0}^{t} a_{ijk} g_i, \tag{II.5}$$

where $a_{ijk} \in K[X_1, \ldots, X_n]$. As we used the division algorithm, we know that

$$\exp(a_{ijk} g_i) \preceq \exp(S(g_j, g_k)) \tag{II.6}$$

for all i, j, k.

We multiply II.5 by $X^{\delta-\gamma_{jk}}$ to obtain

$$X^{\delta-\gamma_{jk}}S(g_j,g_k)=\sum_{i=0}^{t}b_{ijk}g_i, \tag{II.7}$$

where $b_{ijk}=X^{\delta-\gamma_{jk}}a_{ijk}$. Then, by II.6 and Lemma (8.2.)

$$\exp(b_{ijk}g_i)\preceq\exp(X^{\delta-\gamma_{jk}}S(g_j,g_k))\prec\delta. \tag{II.8}$$

Substituting II.7 in into II.4, we get

$$\sum_{m(i)=\delta}LT(h_i)g_i=\sum_{m(i)=\delta}c_{jk}X^{\delta-\gamma_{jk}}S(g_j,g_k)=\sum_{m(i)=\delta}c_{jk}\left(\sum_{i=0}^{t}b_{ijk}g_i\right)=\sum_{m(i)=\delta}\tilde{h}_ig_i.$$

By II.8, we know that for all i,

$$\exp(\tilde{h}_ig_i)\prec\delta.$$

Finally, if we substitute $\sum_{m(i)=\delta}LT(h_i)g_i=\sum_{m(i)=\delta}\tilde{h}_ig_i$ into II.3, we get an expression of f as a polynomial combination of g_i's where all terms have exponent$\prec\delta$. This contradicts the minimality of δ. □

This Buchberger's algorithm is taken from [35].

BUCHBERGERS_ALGORITHM($G*$):

Input: $G*$ basis of an ideal $I\subset K[X_1,\ldots,X_n]$

Result: A Gröbner basis G for I with $G*\subset G$

$G=G*$

$P=\{(f,g)\,|\,f,g\in G, f\neq g\}$

while $P\neq\varnothing$:

Choose $(f,g)\in P$

$P=P-(f,g)$

$s = \mathcal{R}[S(f,g) : G]$

if $s \neq 0$:

$P = P \cup \{(s,f) \mid f \in G\}$

$G = G \cup s$

The algorithm includes the remainder of the S-polynomials that are non-zero. When including a new polynomial, it might happen that some of the others might be removed from the basis, and it is still a Gröbner basis.

Lemma. 10.2.
Let G be a Gröbner basis for the polynomial ideal I. If $p \in G$ such that $LT(p) \in \langle LT(G - \{p\})\rangle$. Then, $G - \{p\}$ is also a Gröbner basis of I.

PROOF. As it is a Gröbner basis, $\langle LT(G)\rangle = \langle LT(I)\rangle$. If $LT(p) \in \langle LT(G - \{p\})\rangle$, then $LT(G - \{p\}) = LT(G)$. Therefore, $G - \{p\}$ is also a Gröbner basis of I. □

If G is a Gröbner basis of an ideal I, then any $G* \subset I$ such that $G \subset G*$ is also a Gröbner basis of I. Therefore, we can simplify a Gröbner basis to obtain a minimal Gröbner basis. The minimal Gröbner basis of an ideal is unique for a given monomial ordering.

Removing from G any p such that $LT(p) \in \langle LT(G - \{p\})\rangle$ and adjusting the leading coefficients to be 1, we get the *minimal Gröbner basis*.

A **minimal Gröbner basis** for a polynomial ideal I is a Gröbner basis G such that:

- $LC(p) = 1$ for all $p \in G$.
- For all $p \in G$, $LT(p) \notin \langle LT(G - \{p\})\rangle$.

Chapter III

Application of Gröbner bases

In this chapter, we see some applications of Gröbner bases to the satisfiability problem. In particular, we will see how to use them to solve the satisfiability problem in propositional logic and the existential fragment of Presburger's theory. Additionally, the application of Gröbner bases to the non-linear real arithmetic can be found in [30, 31].

The content of this chapter has been taken from [8, 27, 25, 4, 12].

11 Algebraic SAT solver

In algebraic SAT solvers, formulas are translated to polynomials so that questions of satisfiability can be answered using algebraic methods. In what follows, we will work with the finite field $\mathbb{F}_2$ and polynomials in the polynomial ring $\mathbb{F}_2[X_1,\ldots,X_n]$.

11.1 From proposition to polynomial

To work with Gröbner basis, we need to express a proposition $P \in \mathcal{L}_p$ as a polynomial $p \in \mathbb{F}_2[X_1,\ldots,X_n]$. The truth values will be represented by the elements of the field $\mathbb{F}_2 = \{0,1\}$ and $p \in \mathbb{F}_2[X_1,\ldots,X_n]$, where the variables $X_1,\ldots,X_n \in \mathbb{F}_2$ are associated with the propositional atoms $A_1,\ldots,A_n \in \mathcal{A}_p$. The calculation of the polynomial associated to a proposition is done by means of θ that we define as follows.

We define $\theta : \mathcal{L}_P \to \mathbb{F}_2[X_1,\dots,X_n]$ to compute the polynomial associated with a proposition:

- $\theta(A_i) = X_i$ for all $A_i \in \mathcal{A}_P$.
- For $C \in \mathcal{C}$,

  $$c(X_1,\dots,X_d) = \theta(C(A_{i_1},\dots,A_{i_d})) = \sum_{(a_1,\dots,a_d)\in\mathbb{F}_2^d} C(a_1,\dots,a_d)G_{a_1}(\theta(A_{i_1}))\cdots G_{a_d}(\theta(A_{i_d})),$$

  where $G_0(P) = P+1$ and $G_1(P) = P$, for each $P \in \mathcal{L}_P$.

Given $P, P_1,\dots,P_d \in \mathcal{L}_P$, we compute the associated polynomial $p \in \mathbb{F}_2[X_1,\dots,X_d]$ recursively,

$$\theta(P(P_1,\dots,P_d)) = p(\theta(P_1),\dots,\theta(P_d)).$$

Example. 11.1.

1. When $C = id$, then $\theta(id(A)) = \sum_{a\in\mathbb{F}_2} id(a)G_a(\theta(A)) = 0G_0(\theta(A)) + 1G_1(\theta(A)) = \theta(A) = X$.

   Given $P \in \mathcal{L}_P$, then $\theta(id(P)) = \theta(P)$.

2. When $C = \neg$, then

   $$\begin{aligned}\theta(\neg(A)) &= \sum_{a\in\mathbb{F}_2} \neg(a)G_a(\theta(A)) = 1G_0(\theta(A)) + 0G_1(\theta(A)) \\ &= \theta(A) + 1 = X + 1.\end{aligned}$$

   Given $P \in \mathcal{L}_P$, then $\theta(\neg(P)) = \theta(P) + 1$.

3. When $C = \wedge$, then

   $$\begin{aligned}\theta(\wedge(A,B)) &= \sum_{(a,b)\in\mathbb{F}_2^2} \wedge(a,b)G_a(\theta(A))G_b(\theta(B)) \\ &= 0G_0(\theta(A))G_0(\theta(B)) + 0G_0(\theta(A))G_1(\theta(B)) + 0G_1(\theta(A))G_0(\theta(B)) \\ &\quad + 1G_1(\theta(A))G_1(\theta(B)) \\ &= \theta(A)\theta(B) = XY.\end{aligned}$$

Given $P_1, P_2 \in \mathcal{L}_P$, then $\theta(P_1 \wedge P_2) = \theta(P_1)\theta(P_2)$.

4. When $C = \vee$ and $P_1, P_2 \in \mathcal{L}_P$, then $\theta(P_1 \vee P_2) = \theta(P_1) + \theta(P_2) + \theta(P_1)\theta(P_2)$.

5. When $C = \rightarrow$ and $P_1, P_2 \in \mathcal{L}_P$, then $\theta(P_1 \rightarrow P_2) = \theta(P_1) + \theta(P_1)\theta(P_2) + 1$.

6. When $C = \leftrightarrow$ and $P_1, P_2 \in \mathcal{L}_P$, then $\theta(P_1 \leftrightarrow P_2) = \theta(P_1) + \theta(P_2) + 1$.

7. When $C = \oplus$ and $P_1, P_2 \in \mathcal{L}_P$, then $\theta(P_1 \oplus P_2) = \theta(P_1) + \theta(P_2)$.

Given this conversion, we can say that $p \in \mathbb{F}_2[X_1, \ldots, X_n]$ is *satisfiable* if there exists a point $(s_1, \ldots, s_n) \in \mathbb{F}_2^n$ such that $p(s_1, \ldots, s_n) = 1$. Thus, we can define satisfiability in terms of polynomials as follows.

Corollary. 11.2.
*Let $p \in \mathbb{F}_2[X_1, \ldots, X_n]$, p is **satisfiable** if there exists $\bar{s} \in \mathbb{F}_2^n$ such that $\overline{p}(\bar{s}) = p(\bar{s}) + 1 = 0$.*

We redefine the satisfiability problem as a set of equations. The existence of common zeros is the answer to whether the formula is satisfiable. Since $X_1, \ldots, X_n \in \mathbb{F}_2$, we use the polynomials $X_i^2 + X_i$ for $i = 1, \ldots, n$, whose roots are $0, 1$. Then, the decision problem is equivalent to whether there exists a solution to the following system of equations:

$$\begin{aligned} p(X_1, \ldots, X_n) &= 0 \\ X_1^2 + X_1 &= 0 \\ &\vdots \\ X_n^2 + X_n &= 0 \end{aligned}$$

11.2 Hilbert's Nullstellensatz

The Hilbert's Nullstellensatz theorem gives a condition to determine whether an ideal has or not a common set of zeros, or if that set is empty.

Theorem. 11.3. (Hilbert's Nullstellensatz)
Let K be an algebraically closed field and $I \subset K[X_1,\dots,X_n]$. Then, $V(I) = \varnothing$ if and only if $I = K[X_1,\dots,X_n]$.

To use this result, we just need that $\mathbb{F}_2$ is algebraically closed. This is not the case, but we will use its algebraic closure $\overline{\mathbb{F}_2}$ along with the polynomials $X_i^2 + X_i$. In this way, we can use the Hilbert's Nullstellensatz theorem. When $I = \langle p, X_1^2 + X_1, \dots, X_n^2 + X_n\rangle$, the variety $V(I) = \varnothing$ if and only if $1 \in I$. This is not true for general ideals $I \subset \mathbb{F}_2[X_1,\dots,X_n]$, as we can see in the next example.

Example. 11.4.
Let's consider $I = \langle X^2 + X + 1\rangle \subset \mathbb{F}_2[X]$. Since $X^2 + X + 1$ does not have a zero in $\mathbb{F}_2$, it follows from the Lemma (5.2.) that $V(I) = \varnothing$, but $1 \notin \langle X^2 + X + 1\rangle$.

Theorem. 11.5.
Given $p \in \mathbb{F}_2[X_1,\dots,X_n]$, and considering $I = \langle p, X_1^2 + X_1, \dots, X_n^2 + X_n\rangle$ in $\mathbb{F}_2[X_1,\dots,X_n]$ and $\overline{I} = \langle p, X_1^2 + X_1, \dots, X_n^2 + X_n\rangle$ in $\overline{\mathbb{F}_2}[X_1,\dots,X_n]$. Then, $V(I) = \varnothing$ if and only if $V(\overline{I}) = \varnothing$.

PROOF. We prove that $V(I) \neq \varnothing \iff V(\overline{I}) \neq \varnothing$, what is equivalent to $V(I) = \varnothing \iff V(\overline{I}) = \varnothing$.

$\Rightarrow$) When $V(I) \neq \varnothing$ there exists $\bar{s} \in \mathbb{F}_2^n$ such that $p(\bar{s}) = 0$ for all $p \in I$. Given $p^* \in \overline{I}$, by the definition of an ideal, it can be written as $p^* = h_0 \cdot p + \sum_{i=1}^n h_i \cdot (X_i^2 + X_i)$, where $h_i \in \overline{\mathbb{F}_2}[X_1,\dots,X_n]$. It follows that $p^*(\bar{s}) = \sum_{i=0}^n h_i \cdot 0 = 0$ and since $\bar{s} \in \mathbb{F}_2^n \subset \overline{\mathbb{F}_2}^n$, then $\bar{s} \in V(\overline{I})$.

$\Leftarrow$) Suppose that $V(I) = \varnothing$ while $V(\overline{I}) \neq \varnothing$. By the definition of varieties, it follows that each $s \in V(\overline{I})$ is an element of $\overline{\mathbb{F}_2}^n \setminus \mathbb{F}_2^n$. Since $X_1^2 + X_1, \dots, X_n^2 + X_n$ are contained in $\overline{I}$, a zero $\bar{s}$ must satisfy $X_i^2 + X_i = 0$ for each i. This implies that each $s_i \in \{0, 1\}$, what contradicts the fact that $\bar{s} \notin \mathbb{F}_2^n$. Therefore, $V(I) \neq \varnothing$. □

Corollary. 11.6.
Given the ideal $I \subset \mathbb{F}_2[X_1,\dots,X_n]$ *such that* $I = \langle p, X_1^2 + X_1, \dots, X_n^2 + X_n\rangle$ *for* $p \in \mathbb{F}_2[X_1,\dots,X_n]$. *Then,* $V(I) = \varnothing$ *if and only if* $1 \in I$.

PROOF.

$\Leftarrow$) Trivial. Since $1 \in I$, it does not have any zero.

$\Rightarrow$) From Theorem (11.3.) and Theorem (11.5.), it follows that $V(I) = \varnothing \Leftrightarrow V(\bar{I}) = \varnothing \Leftrightarrow 1 \in \bar{I}$.

Since $I = \bar{I} \cap \mathbb{F}_2[X_1,\dots,X_n]$, if $1 \in \bar{I}$ then $1 \in I$. □

The last result gives a way to check satisfiability.

Theorem. 11.7.
Given $p \in \mathbb{F}_2[X_1,\dots,X_n]$, *we say that* p *is* ***satisfiable****, if and only if,* $1 \notin I = \langle p+1, X_1^2 + X_1, \dots, X_n^2 + X_n\rangle$.

11.3 Adapted Buchberger criterion

Returning to Buchberger's criterion 10.1., but now applied to the satisfiability problem. It turns out that the S-polynomials $S(X_i^2 + X_i, X_j^2 + X_j)$ have zero remainder when dividing by G, which allows us to save some computations in the Buchberger's algorithm.

Proposition. 11.8.
Let G *be the ordered set* $\{X_1^2 + X_1, \dots, X_n^2 + X_n\}$. *Then,* $\mathcal{R}[S(X_i^2 + X_i, X_j^2 + X_j) : G] = 0$ *for* $1 \leqslant i, j \leqslant n$.

PROOF. Using the division algorithm to divide $S(X_i^2+X_i, X_j^2+X_j) = X_j^2X_i + X_i^2X_j$ by the ordered G, we get $S(X_i^2+X_i, X_j^2+X_j) = X_i(X_j^2+X_j) + X_j(X_i^2+X_i) + 0$, where $r = 0$. □

Corollary. 11.9.
Let $I = \langle p, X_1^2+X_1, \ldots, X_n^2+X_n\rangle$. Then, $G = \{p, X_1^2+X_1, \ldots, X_n^2+X_n\}$ is a Gröbner basis if and only if, for all i, the remainder on division of $S(p, X_i^2+X_i)$ by the ordered set $\{X_1^2+X_1, \ldots, X_n^2+X_n, p\}$ is zero.

PROOF. Trivial if we bear in mind the Buchberger's criterion (10.1.) and Proposition (11.8.) above. □

11.4 Algorithm

In this section, we define some methods to check satisfiability using Gröbner bases together with the adapted Buchberger's criterion.

SAT_BASED_ON_GROEBNER_BASIS(F):

Input: $P \in \mathscr{L}_P$

Result: Whether the formula is satisfiable, `true` or `false`

Translate P to a polynomial p

Compute $\overline{p} = p + 1$

return ADAPTED_BUCHBERGERS_ALGORITHM($\overline{p}$)

The next algorithm implements the Buchberger algorithm with the simplification of the Buchberger's criterion and ends when it finds an S-polynomial whose remainder is zero when dividing by G. It returns `false` if it finds such S-polynomial since in such case $1 \notin G$ and there is no solution to the system of equations, or `true` in case it does not find it.

ADAPTED_BUCHBERGERS_ALGORITHM(p):

Input: $p \in \mathbb{F}_2[X_1,\ldots,X_n]$

Result: A Gröbner basis of $\{p, X_1^2+X_1,\ldots,X_n^2+X_n\}$

$G = \{p, X_1^2+X_1,\ldots,X_n^2+X_n\}$

$P = \{(p, X_i^2+X_i) \mid i = 1,\ldots,n\}$

while $P \neq \emptyset$:

Choose $(p,g) \in P$

$P = P-(p,g)$

$s = \mathcal{R}[S(p,g) : G]$

if $s = 1$:

return `false`

if $s \neq 0$:

$P = P \cup \{(s,p) \mid p \in G\}$

$G = G \cup s$

return `true`

In this project, a simple implementation of an algebraic SAT solver has also been carried out, although the adaptation of the Buchberger's criterion is not used. This can be found later in the report.

12 Existential fragment of Presburger's arithmetic

Given a formula of the existential fragment of Presburger's theory in the form

$$D_1 \vee \cdots \vee D_m = (P_{11} \wedge \cdots \wedge P_{1n_1}) \vee \cdots \vee (P_{m1} \wedge \cdots \wedge P_{mn_m}),$$

where each predicate P_{ij} is a lineal equation of the form $a_1Z_1+\cdots+a_nZ_n = b$ for $a_1,\cdots a_n, b \in \mathbb{N}$ and $Z_1,\ldots,Z_n$ take values in the natural numbers.

For the formula to be satisfiable, we need at least one of the disjunctions D_i to be true, but this only occurs if there exists an i such that the system of equations represented by D_i has a solution. Therefore, we need to check whether there exists a solution $\bar{s} = (s_1, \ldots, s_m) \in \mathbb{N}^m$ such that

$$A\bar{s} = b, \text{ where } A \in \mathbb{N}^{n \times m} \text{ and } b \in \mathbb{N}^n.$$

To do so, we will use a K-algebra homomorphism.

A K**-algebra homomorphism** is a ring homomorphism

$$\phi : K[Y_1, \ldots, Y_m] \to K[X_1, \ldots, X_n]$$

which is a K-vector space linear transformation. These maps are uniquely determined by

$$\phi : Y_i \mapsto F_i,$$

where $F_i \in K[X_1, \ldots, X_n]$ with $1 \leqslant i \leqslant m$.

A system as follows is given, where $a_{ij}, b_j \in \mathbb{N}$.

$$\begin{aligned} a_{11}Z_1 + \cdots + a_{1m}Z_m &= b_1 \\ a_{21}Z_1 + \cdots + a_{2m}Z_m &= b_2 \\ &\vdots \\ a_{n1}Z_1 + \cdots + a_{nm}Z_m &= b_n \end{aligned} \tag{III.1}$$

To determine if there is a solution, we translate this linear equations into multivariate polynomials by assigning a variable $X_1, \ldots, X_n$ to each constraint. Then for each equation we obtain

$$X_i^{a_{i1}Z_1 + \cdots + a_{im}Z_m} = X_i^{b_i}.$$

As we want all of this equation to be met, we can multiply them to get

$$X_1^{a_{11}Z_1 + \cdots + a_{1m}Z_m} \ldots X_n^{a_{n1}Z_1 + \cdots + a_{nm}Z_m} = X_1^{b_1} \ldots X_n^{b_n}.$$

If we rearrange the multiplications, it is equivalent to

$$(X_1^{a_{11}} \dots X_n^{a_{n1}})^{Z_1} \dots (X_1^{a_{1m}} \dots X_n^{a_{nm}})^{Z_m} = X_1^{b_1} \dots X_n^{b_n}. \tag{III.2}$$

At this point, for the polynomial rings $K[Y_1,\dots,Y_n]$ and $K[X_1,\dots,X_n]$ we can define the polynomial map

$$\begin{aligned} \phi : K[Y_1,\dots,Y_n] &\to K[X_1,\dots,X_n] \\ Y_j &\mapsto F_j = X_1^{a_{11}} \dots X_n^{a_{n1}}, \end{aligned} \tag{III.3}$$

and rewrite the equation III.2 as

$$(\phi(Y_1))^{Z_1} \dots (\phi(Y_m))^{Z_m} = \phi(Y_1^{Z_1} \dots Y_m^{Z_m}) = X_1^{b_1} \dots X_n^{b_n}.$$

Then, a solution $(s_1,\dots,s_m)$ satisfies $\phi(Y_1^{s_1} \dots Y_m^{s_m}) = X_1^{b_1} \dots X_n^{b_n}$.

Lemma. 12.1.
Assuming that $a_{ij}, b_j \in \mathbb{N}$. Then, there exists a solution $\bar{s} = (s_1,\dots,s_m) \in \mathbb{N}^m$ of the system III.1 if and only if $X_1^{b_1} \dots X_n^{b_n}$ is the image under ϕ of some $Y_1^{s_1} \dots Y_m^{s_m} \in K[Y_1,\dots,Y_m]$. Moreover, if $X_1^{b_1} \dots X_n^{b_n} = \phi(Y_1^{s_1} \dots Y_m^{s_m})$ then $\bar{s}$ is a solution of the system.

In the following result, it is shown that given a K-algebra homomorphism like the one we have in III.3, the kernel of ϕ is equals to $\langle Y_1 - F_1,\dots,Y_m - F_m\rangle \cap K[Y_1,\dots,Y_m]$. By the first theorem of isomorphisms for rings, there exists an isomorphism of rings

$$K[Y_1,\dots,Y_m]/\mathrm{Ker}(\phi) \to \mathrm{Im}(\phi)$$

defined by

$$g + \mathrm{Ker}(\phi) \mapsto \phi(g).$$

Theorem. 12.2.
Let $\phi : K[Y_1,\dots,Y_m] \to K[X_1,\dots,X_n]$ as in III.3 and considering $H = \langle Y_1 - F_1,\dots,Y_m - F_m\rangle$. Then, $\mathrm{Ker}(\phi) = H \cap K[Y_1,\dots,Y_m]$.

PROOF.

$\subset$) If $g \in \mathrm{Ker}(\phi) \subset K[Y_1, \ldots, Y_m]$, then

$$g = \sum_{\alpha} c_\alpha Y_1^{\alpha_1} \cdots Y_m^{\alpha_m},$$

where $c_\alpha \in K$, $\alpha = (\alpha_1, \ldots, \alpha_m) \in \mathbb{N}^m$ and only finitely many c_α's are non-zero. As $g \in \mathrm{Ker}(\phi)$, then $\phi(g(Y_1, \ldots, Y_m)) = g(F_1, \ldots, F_m) = 0$. Hence,

$$\begin{aligned} g &= g - g(F_1, \ldots, F_m) \\ &= \sum_{\alpha} c_\alpha Y_1^{\alpha_1} \cdots Y_m^{\alpha_m} - \sum_{\alpha} c_\alpha F_1^{\alpha_1} \cdots F_m^{\alpha_m} \\ &= \sum_{\alpha} c_\alpha (Y_1^{\alpha_1} \cdots Y_m^{\alpha_m} - F_1^{\alpha_1} \cdots F_m^{\alpha_m}). \end{aligned}$$

Using the next lemma which is proved in [4], we can conclude that $g \in H \cap K[Y_1, \ldots, Y_m]$.

Lemma. 12.3.
Let R be a commutative ring and $a_1, \ldots, a_n, b_1, \ldots, b_n \in R$. Then $a_1 \cdots a_n - b_1 \cdots b_n$ is in the ideal $\langle a_1 - b_1, \ldots, a_n - b_n \rangle$.

$\supset$) If $g \in H \cap K[Y_1, \ldots, Y_m]$, then

$$g(Y_1, \ldots, Y_m) = \sum_{i=1}^{m} (Y_i - F_i(X_1, \ldots, X_n)) h_i,$$

where $h_i \in K[Y_1, \ldots, Y_m, X_1, \ldots, X_n]$. Hence,

$$\phi(g) = g(F_1, \ldots, F_m) = \sum_{i=1}^{m} (F_i(X_1, \ldots, X_n) - F_i(X_1, \ldots, X_n)) h_i = 0.$$

Therefore, $g \in \mathrm{Ker}(\phi)$. □

In fact, we know that not only does $\phi(g + \mathrm{Ker}(\phi)) = \phi(g)$ occur, but $\phi(g + H) = \phi(g)$ occurs as well. As the Lemma (12.1.) says, for there to be a solution of III.1, it is necessary that $X_1^{b_1} \ldots X_n^{b_n} \in \mathrm{Im}(\phi)$. This theorem gives a way to determine when a polynomial is in the image of ϕ.

Theorem. 12.4.
If we consider the ideal $H = \langle Y_1 - F_1, \ldots, Y_m - F_m \rangle \subset K[Y_1, \ldots, Y_m, X_1, \ldots, X_n]$ *and* G *a reduced Gröbner basis of* H *with respect to an elimination order with the X's priority larger than Y's. Then* $f \in K[X_1, \ldots, X_n]$ *is in* $\mathrm{Im}(\phi)$ *if and only if there exists* $h \in K[Y_1, \ldots, Y_m]$ *such that* $\mathcal{R}[f : G] = h$.

PROOF.

$\Rightarrow$) Suppose that $f \in K[X_1, \ldots, X_n]$ is in $\mathrm{Im}(\phi)$, then there exists $h \in K[Y_1, \ldots, Y_m]$ such that $f = \phi(h)$. Applying ϕ, we obtain $f = \phi(h(Y_1, \ldots, Y_m)) = h(F_1, \ldots, F_m)$. Thus,

$$f(X_1, \ldots, X_n) - h(Y_1, \ldots, Y_m) = h(F_1, \ldots, F_m) - h(Y_1, \ldots, Y_m) \in K[Y_1, \ldots, Y_m, X_1, \ldots, X_n].$$

We can observe that $h(F_1, \ldots, F_m) - h(Y_1, \ldots, Y_m)$ has pairs of monomials which are of the form $F_1^{\alpha_1} \cdots F_m^{\alpha_m} - Y_1^{\alpha_1} \cdots Y_m^{\alpha_m}$ and by Lemma (12.3.) $f(X_1, \ldots, X_n) - h(Y_1, \ldots, Y_m) \in H$. Therefore, we have $\mathcal{R}[f(X_1, \ldots, X_n) - h(Y_1, \ldots, Y_m) : G] = 0$. Then,

$$\mathcal{R}[f(X_1, \ldots, X_n) : G] = \mathcal{R}[h(Y_1, \ldots, Y_m) : G] \tag{III.4}$$

and by Proposition (9.3.) we know that the remainder is unique.

Since the X's are larger than the Y's, $h \in K[Y_1, \ldots, Y_m]$ can only be divided by polynomials whose leading term only contains Y's. Besides, $\mathcal{R}[h : G]$ does not contain X's, so $\mathcal{R}[h : G] \in K[Y_1, \ldots, Y_m]$. By III.4 and Proposition (9.3.), we conclude that $\mathcal{R}[f(X_1, \ldots, X_n) : G] \in K[Y_1, \ldots, Y_m]$.

$\Leftarrow$) Suppose that for $f \in K[X_1, \ldots, X_n]$, there exists $h \in K[Y_1, \ldots, Y_m]$ such that $\mathcal{R}[f : G] = h$. Then, $\mathcal{R}[f - h : G] = 0$ and $f - h \in H$. Therefore,

$$f(X_1, \ldots, X_n) - h(Y_1, \ldots, Y_m) = \sum_{i=1}^{m} (Y_i - F_i(X_1, \ldots, X_n)) q_i(Y_1, \ldots, Y_m, X_1, \ldots, X_n).$$

Applying the homomorphism ϕ, we get

$$\phi(f(X_1, \ldots, X_n) - h(Y_1, \ldots, Y_m)) = \phi\left(\sum_{i=1}^{m} (Y_i - F_i(X_1, \ldots, X_n)) q_i(Y_1, \ldots, Y_m, X_1, \ldots, X_n)\right)$$

$$\phi(f(X_1, \ldots, X_n)) - \phi(h(Y_1, \ldots, Y_m)) = \sum_{i=1}^{m} \phi((Y_i - F_i(X_1, \ldots, X_n)) q_i(Y_1, \ldots, Y_m, X_1, \ldots, X_n))$$

$$f(X_1,\dots,X_n)-h(F_1,\dots,F_m)$$
$$=\sum_{i=1}^{m}(F_i(X_1,\dots,X_n)-F_i(X_1,\dots,X_n))q_i(F_1,\dots,F_m,X_1,\dots,X_n)=0.$$

As $f(X_1,\dots,X_n)-h(F_1,\dots,F_m)=0$, then $f(X_1,\dots,X_n)=h(F_1,\dots,F_m)=\phi(h(Y_1,\dots,Y_m))$. Finally, we conclude that $f\in \mathrm{Im}(\phi)$. □

Therefore, there exists a solution of the system III.1 if and only if $\mathcal{R}[X_1^{b_1}\cdots X_n^{b_n} : G] = Y_1^{s_1}\cdots Y_m^{s_m}$ for some $(s_1,\dots,s_m)\in\mathbb{N}^m$.

Example. 12.5.
We want to know whether there exists a solution of the system

$$Z_1+Z_2=3$$
$$Z_1+2Z_2=4.$$

Then, for each equation the variables X_1, X_2 are introduced,

$$X_1^{Z_1+Z_2}=X_1^3$$
$$X_2^{Z_1+2Z_2}=X_2^4.$$

After combining all and rearranging, we get $(X_1X_2)^{Z_1}(X_1X_2^2)^{Z_2}=X_1^3X_2^4$.

Now, the K-algebra homomorphism can be defined as

$$\phi : K[Y_1,Y_2]\to K[X_1,X_2]$$
$$Y_1\mapsto X_1X_2$$
$$Y_2\mapsto X_1X_2^2.$$

Then, $H=\langle Y_1-X_1X_2, Y_2-X_1X_2^2\rangle$. The result of computing the Gröbner basis of H with a lex order where $X_1>X_2>Y_1>Y_2$ is $G=\{g_1,g_2,g_3\}$ where

$$g_1=X_1X_2-Y_1$$
$$g_2=X_1Y_2-Y_1^2$$
$$g_3=X_1Y_1-Y_2.$$

Then, $\mathcal{R}[X_1^3X_2^4 : G]=Y_1^2Y_2$. Therefore, the system has a solution and $(2,1)$ is a solution.

It is expected that if a power product $X_1^{b_1}\cdots X_n^{b_n} \in K[X_1,\ldots,X_n]$ is the image under ϕ of some $h \in K[Y_1,\ldots,Y_m]$, then h is also a power product $Y_1^{s_1}\cdots Y_m^{s_m} \in K[Y_1,\ldots,Y_m]$. In this lemma, we will see that this always occurs.

Lemma. 12.6.
Taking the elements defined so far. If $X_1^{b_1}\cdots X_n^{b_n} \in \mathrm{Im}(\phi)$*, then it is the image of a power product* $Y_1^{s_1}\cdots Y_m^{s_m}$.

PROOF. By Theorem (12.4.),

$$X_1^{b_1}\cdots X_n^{b_n} \in \mathrm{Im}(\phi) \iff \mathcal{R}[X_1^{b_1}\cdots X_n^{b_n} : G] = h \in K[Y_1,\ldots,Y_m].$$

Moreover, $\phi(h) = X_1^{b_1}\cdots X_n^{b_n}$.

On the other hand, any combination of two polynomials that are sum of two power products is a sum of two power products, then the polynomials in G are polynomials that are sum of two power products.

Since, any combination of a power product and a polynomial which is the sum of two power products gives as a result a power product, then h is a power product. □

Example. 12.7.
We want to know whether there exists a solution of the system of linear equations

$$\begin{aligned} 3Z_1 + 2Z_2 + Z_3 + Z_4 &= 10 \\ 4Z_1 + Z_2 + Z_3 &= 5. \end{aligned}$$

Then, for each equation the variables X_1, X_2 are introduced,

$$\begin{aligned} X_1^{3Z_1+2Z_2+Z_3+Z_4} &= X_1^{10} \\ X_2^{4Z_1+Z_2+Z_3} &= X_2^{5}. \end{aligned}$$

We combine all and rearrange them to get

$$(X_1^3X_2^4)^{Z_1}(X_1^2X_2)^{Z_2}(X_1X_2)^{Z_3}(X_1)^{Z_4} = X_1^{10}X_2^5.$$

Now, the K-algebra homomorphism can be defined as

$$\begin{aligned}\phi : K[Y_1, Y_2, Y_3, Y_4] &\to K[X_1, X_2] \\ Y_1 &\mapsto X_1^3 X_2^4 \\ Y_2 &\mapsto X_1^2 X_2 \\ Y_3 &\mapsto X_1 X_2 \\ Y_4 &\mapsto X_1.\end{aligned}$$

Then, $H = \langle Y_1 - X_1^3 X_2^4, Y_2 - X_1^2 X_2, Y_3 - X_1 X_2, Y_4 - X_1 \rangle$. The result of computing the Gröbner basis of H with a lex order where $X_1 > X_2 > Y_1 > Y_2 > Y_3 > Y_4$ is $G = \{g_1, g_2, g_3, g_4, g_5\}$ where

$$\begin{aligned}g_1 &= X_1 - Y_4 \\ g_2 &= X_2 Y_3^3 - Y_1 \\ g_3 &= X_2 Y_4 - Y_3 \\ g_4 &= Y_1 Y_4 - Y_3^4 \\ g_5 &= Y_2 - Y_3 Y_4.\end{aligned}$$

Then, $\mathcal{R}[X_1^{10} X_2^5 : G] = Y_4^5 Y_3^5$. Therefore, the system has a solution and $(0, 0, 5, 5)$ is a solution.

Conclusions and future directions

In this thesis, we have introduced propositional and first order logic. We have proved Herbrand's expansion theorem which together with the tiling problem has helped us to show the undecidability of the satisfiability problem in first order logic, as well as NP-completeness in propositional logic. We have defined first order theories and given some examples of them. We then introduced Gröbner bases and applied them to SAT and Presburger's theory of arithmetic.

For a more complete overview of the resolution of the existential fragment of Presburger's theory, it would have been good to review some other references like [6, 20], due to the relation of Presburger's theory to semigroups and the integer linear programming problem. It would also have been nice to discuss the resolution of the satisfiability problem of the theory of reals that we can find in the references [30, 31], since it also use Gröbner bases.

A simple implementation of the SAT algebraic approach has been made. To improve this implementation, the translation from proposition to polynomial could be improved, as seen in this article [26] where formulas in conjunctive normal form are translated. It might also be interesting to have a look into some of the improvements that have now been seen to the Buchberger's algorithm [37, 33, 22, 21, 11]. There might already be Gröbner bases calculation algorithms efficiently adapted to SAT [40]. Moreover, a more thoughtful reading of the book [12] might be advisable. In this book several algebraic approaches to solving SAT are presented, these are called *Nullstellenstaz*, *Polynomial Calculus* and *Polynomial Calculus Resolution*.

APPLICATIONS: Many real-world problems have been formalized as SAT decision problems, so it is used in planning and scheduling problems, as well as in theorem proving, software and hardware verification and cryptography among others.

Since the boolean satisfiability problem is NP-complete, then any problem in NP can be modelled on propositional logic. More precisely, any NP problem can be coded into propositional logic in polynomial time and then be solved by a SAT solver. During the last two decades,

there has been a sustained improvement in SAT solving technology. Given this fact, being able to encode a problem into SAT is very likely to result in a practical solution.

There are a few techniques a leading contemporary SAT solver is based on: existential quantification, sound and complete inference rules and systematic search in the space of truth assignments. DPLL algorithm form the basis of most modern SAT solvers. Due to the many refinements these algorithms have experimented, they are the most efficient algorithms thus far. The most promising have been Conflict-Driven Clause Learning SAT solvers, they are based on a combination of those three techniques.

We have used Gröbner bases to solve the satisfiability problem in propositional logic. Algebraic SAT solvers seem not to have been very successful due to the breakthrough of CDCL solvers. The reason may be that algebraic SAT solvers has not yet been sufficiently explored. In addition, the computation of Gröbner bases is known to be an EXPSPACE-complete problem [10]. It is expected that for this type of SAT solvers to be successful, a cut-off for the Gröbner basis algorithm must be found, since it allows not only to decide the satisfiability but also to enumerate the satisfying assignments. Despite this fact, for some cases, it seems to be a good approach as stated in the book [12]. *March* [24] and *CryptoMiniSat* [39] are a few examples of algebraic SAT solvers.

Despite the complexity of SAT and SMT solving, in practice there are SMT solvers that have shown the feasibility of practical automated decision procedures for this problem, this has helped to solve complex and interesting problems. A few examples of SMT solvers developed in the industry are *cvc5* [9] and *Z3* [34].

Bibliography

[1] Erika Ábrahám. "Building bridges between symbolic computation and satisfiability checking". In: *Proceedings of the 2015 ACM on International Symposium on Symbolic and Algebraic Computation*. 2015, pp. 1–6.

[2] Erika Ábrahám et al. "Satisfiability checking and symbolic computation". In: *ACM Communications in Computer Algebra* 50.4 (2017), pp. 145–147.

[3] Erika Ábrahám et al. "SC2: Satisfiability checking meets symbolic computation". In: *Intelligent Computer Mathematics: Proceedings CICM* 9791 (2016), pp. 28–43.

[4] William W Adams and Philippe Loustaunau. *An introduction to Gröbner bases*. Vol. 3. American Mathematical Society, 1994.

[5] Dennis S Arnon, George E Collins, and Scott McCallum. "Cylindrical algebraic decomposition I: The basic algorithm". In: *SIAM Journal on Computing* 13.4 (1984), pp. 865–877.

[6] Abdallah Assi, Marco D'Anna, and Pedro A Garcıa-Sánchez. *Numerical semigroups and applications*. Vol. 3. Springer Nature, 2020.

[7] Franz Baader and Tobias Nipkow. *Term rewriting and all that*. Cambridge university press, 1999.

[8] PW Bakker. "An algebraic approach to the Boolean Satisfiability Problem". MA thesis. University of Groningen, 2016.

[9] Haniel Barbosa et al. "cvc5: A Versatile and Industrial-Strength SMT Solver". In: *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer. 2022, pp. 415–442.

[10] Magali Bardet. "On the complexity of a Grobner Basis algorithm". In: *Algorithms Seminar 2002-2004*. 2005.

[11] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. "On the complexity of the F5 Gröbner basis algorithm". In: *Journal of Symbolic Computation* 70 (2015), pp. 49–70.

[12] Armin Biere, Marijn Heule, and Hans van Maaren. *Handbook of satisfiability - 2nd Edition*. Vol. 336. IOS press, 2020.

[13] Egon Börger, Erich Grädel, and Yuri Gurevich. *The classical decision problem*. Springer Science & Business Media, 2001.

[14] Aaron R Bradley and Zohar Manna. *The calculus of computation: decision procedures with applications to verification*. Springer Science & Business Media, 2007.

[15] Curtis Bright, Ilias Kotsireas, and Vijay Ganesh. "When satisfiability solving meets symbolic computation". In: *Communications of the ACM* 65.7 (2022), pp. 64–72.

[16] Stephen A Cook. "The complexity of theorem-proving procedures". In: *Proceedings of the third annual ACM symposium on Theory of computing*. 1971, pp. 151–158.

[17] David Cox, John Little, and Donal OShea. *Ideals, varieties, and algorithms: an introduction to computational algebraic geometry and commutative algebra*. Springer Science & Business Media, 2013.

[18] O. Strichman D. Kroening. *Decision Procedures: An Algorithmic Point of View - Basic Concepts and Background*. 2007. URL: http://www.decision-procedures.org/slides/basics-2x3.pdf.

[19] James H Davenport et al. *Symbolic computation and satisfiability checking*. 2020.

[20] Jesús A De Loera, Raymond Hemmecke, and Matthias Köppe. *Algebraic and geometric ideas in the theory of discrete optimization*. SIAM, 2012.

[21] Jean Charles Faugere. "A new efficient algorithm for computing Gröbner bases without reduction to zero (F 5)". In: *Proceedings of the 2002 international symposium on Symbolic and algebraic computation*. 2002, pp. 75–83.

[22] Jean-Charles Faugere. "A new efficient algorithm for computing Gröbner bases (F4)". In: *Journal of pure and applied algebra* 139.1-3 (1999), pp. 61–88.

[23] Jean Gallier and Jocelyn Quaintance. "Introduction to the Theory of Computation Computability, Complexity, And the Lambda Calculus Some Notes for CIS262". In: (2020).

[24] Marijn JH Heule and Hans Van Maaren. "March_dl: Adding adaptive heuristics and a new branching strategy". In: *Journal on Satisfiability, Boolean Modeling and Computation* 2.1-4 (2006), pp. 47–59.

[25] M Hoekstra. "Grobner bases and Graver bases used in integer programming". MA thesis. Faculty of Science and Engineering, 2013.

[26] Jan Horáček and Martin Kreuzer. "On conversions from CNF to ANF". In: *Journal of Symbolic Computation* 100 (2020), pp. 164–186.

[27] Pascual Jara. *Lecture notes in basic commutative algebra*. 2020.

[28] Pascual Jara and Evangelina Santos. *Lecture notes in Gröbner bases*. 2020.

[29] Evangelina Santos Jesús Miranda. *Lecture notes in logic and discrete methods*. 2011.

[30] Sebastian Junges et al. "On Gröbner bases in SMT-compliant decision procedures". PhD thesis. Bachelor's thesis. RWTH Aachen University, 2012.

[31] Sebastian Junges et al. "On Gröbner bases in the context of satisfiability-modulo-theories solving over the real numbers". In: *International Conference on Algebraic Informatics*. Springer. 2013, pp. 186–198.

[32] Harry R Lewis and Christos H Papadimitriou. "Elements of the Theory of Computation - 1st Edition". In: *ACM SIGACT News* 29.3 (1998), pp. 62–78.

[33] Rusydi H Makarim and Marc Stevens. "M4GB: an efficient Gröbner-basis algorithm". In: *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation*. 2017, pp. 293–300.

[34] Leonardo de Moura and Nikolaj Bjørner. "Z3: An efficient SMT solver". In: *International conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer. 2008, pp. 337–340.

[35] Thanh Hung Nguyen. "Combinations of Boolean Gröbner bases and SAT solvers". PhD thesis. Technische Universität Kaiserslautern, 2014.

[36] Tobias Nipkow. *Lecture notes in logic*. 2021. URL: https://www21.in.tum.de/teaching/logic/SS21/index.html.

[37] Dylan Peifer. "Selection Strategies in Buchberger's Algorithm". In: (2018).

[38] Uwe Schöning. *Logic for computer scientists*. Springer Science & Business Media, 2008.

[39] Mate Soos, Karsten Nohl, and Claude Castelluccia. "Extending SAT Solvers to Cryptographic Problems". In: *Theory and Applications of Satisfiability Testing - SAT 2009, 12th International Conference, SAT 2009, Swansea, UK, June 30 - July 3, 2009. Proceedings*. Ed. by Oliver Kullmann. Vol. 5584. Lecture Notes in Computer Science. Springer, 2009, pp. 244–257. DOI: 10.1007/978-3-642-02777-2_24. URL: https://doi.org/10.1007/978-3-642-02777-2%5C_24.

[40] Quoc-Nam Tran. "A p-space algorithm for groebner bases computation in boolean rings". In: *Proceedings of World Academy of Science, Engineering and Technology*. Vol. 35. Citeseer. 2008, pp. 495–501.

Web References:

Note[1]

- SAT Live!
- SMT-LIB
- Satisfiability Checking and Symbolic Computation
- Symbolic Computation Techniques in SMT Solving: Mathematical Beauty Meets Efficient Heuristics
- Lecture videos in logic
- Symbolic computation – Wikipedia
- Real closed field – Wikipedia
- Faugère's F4 and F5 algorithms – Wikipedia

[1] Links are enabled in digital format.